Nanyang Technological University

Effective Techniques for Association Rule Mining and Associative Classification

Do Tien Dung

A thesis submitted to the
Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

School of Computer Engineering

2007

Abstract

Association rule mining is one of the major data mining techniques and perhaps the most common form of local-pattern discovery in unsupervised learning systems. Conventionally, an association rule is considered based mainly on the support and confidence measures which represent the usefulness and certainty of the rule respectively. However, the number of rules mined under the support-confidence framework is often too large and many of them are uninteresting. And the association rule mining process is computationally expensive as it requires exhaustive search in the huge space of possible rules. Apart from pattern discovery, association rules can also be used for data classification. Associative classification takes advantage of association rule mining in discovering a set of rules that can accurately generalize a dataset. Associative classification can achieve higher accuracy in comparison with other classification approaches. However, similar to association rule mining, the rule discovery process in associative classification is often complex and computationally expensive. And the set of association rules mined using the association rule mining framework may not be effective for direct employment by a classification task.

This research aims to investigate effective techniques for enhancing association rule mining in both rule representation and mining algorithm, and discovering association rules for associative classification. As a result, we have developed a number of new techniques for association rule mining and associative classification. The contributions of this research are listed as follows:

- A new type of constraint for association rules called category-based constraint is proposed. In addition, an Apriori-based algorithm called Category-based Apriori algorithm (Apriori^{CB}) is also proposed for mining association rules with the category-based constraint. The proposed category-based constraint and Apriori^{CB} algorithm can be used for mining interesting rules from datasets using category patterns. The Apriori^{CB} algorithm is efficient as it can reduce considerably the search space of rules when mining interesting rules.

- Effective measures for association rule mining are investigated and analyzed. In particular, the relative confidence of implication relationship, which is captured by the certainty factor measure, is able to reflect truthfully the implication relations of association rules. When relative confidence is used instead of confidence for association rule mining, it can reduce significantly the number of mined rules while retaining the most interesting ones. This is especially useful for mining datasets with items that are unevenly distributed such as textual datasets.

- A new approach called associative feature selection is proposed for unsupervised feature selection from textual datasets. The proposed approach is based on associative relations among items of datasets to compute relevant features. It can be used for data preprocessing of textual datasets, which are often large and contain many irrelevant features.

- A multiple-step rule discovery approach for associative classification called Mstep-AC is proposed. By focusing on discovering rules for data samples that might be misclassified, the proposed approach can achieve better associative classification accuracy while maintaining the rule discovery process at a level of complexity comparable with conventional AC approaches.

- An approach called AIS-AC, based on the Artificial Immune System, is proposed. The proposed approach aims to discover association rules effectively from large datasets for associative classification. Instead of massively searching for all possible association rules, the proposed approach only finds a subset of association rules that is suitable for effective associative classification in an evolutionary manner. The proposed approach is efficient in tackling the computational complexity problem of the huge search space of rules.

Acknowledgement

First of all, I would like to express my sincere appreciation to my supervisor, Dr. Hui Siu Cheung, for his encouragement, support, and guidance. His vision and knowledge play an important role in this research work. I would also like to thank Dr. Alvis Fong for his helpful comments and suggestions to my research.

I am greatly indebted to the School of Computer Engineering, Nanyang Technology University for providing me the scholarship for carrying out this research. My gratitude also goes to the staff of the Database Technology Laboratory, Teo Choo Eng and Eng Hui Fang, for their friendly supports.

I am thankful to my friends in Singapore including Khai, Vu, Hoang, Tho, Binh, Tam and many others, who make me feel at home thousands of miles away from my country. Last, and the most important, this research would not be materialized without love and encouragement from my family, the big one and the small one.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The rapid development of digital data acquisition and storage technology has resulted in the growth of huge databases. The ability to understand these large and complex databases is increasingly important to virtually all fields of business, science and engineering. The entire process of applying a computer-based methodology, including new techniques, for the discovery of knowledge from data is called *data mining* [Kan03]. Data mining involves the analysis of a large amount of data to find knowledge that is nontrivial and previously unknown [CHY96, HK01]. The knowledge derived through a data mining process is often referred to as *models* or *patterns*. Examples of data patterns include linear equations, rules, clusters, graphs and tree structures.

Two primary tasks of data mining are description and prediction [FU96, Dun03]. *Description* involves finding patterns that describe data, whereas *prediction* produces new and nontrivial information to predict unknown or future data. To mine knowledge from data, a wide range of techniques can be used. For example, *clustering* and *summarization* can be used for descriptive tasks [HL99, JMF99], whereas *classification* and *regression* can be used for predictive tasks [CM98, BH99, HSM01].

Association rule mining (ARM) is a data mining technique which discovers interesting rules frequently observed in a given database. These rules can be used to describe the data structure and relations among the data items in the database. Association rules may also be used for data prediction as they show the tendency from some data items to others. Based on association rules, Liu *et al.* [LHM98] introduced a new classification approach called *associative classification* (AC) which uses association rules for classification. Associative classification can achieve high accuracy in comparison with other classification approaches [LHM98, LMW00, LHP01].

In this research, we focus on investigating association rule mining and associative classification. Before stating the objectives of this research in Section 1.3, the next two sections give a brief review of association rule mining and associative classification. Then, Section 1.4 summarizes the contributions of this research. Finally, Section 1.5 presents the organization of the thesis.

1.1 Association Rule Mining

Association rule mining [AIS93, AS94] is one of the major data mining techniques and perhaps the most common form of local-pattern discovery in unsupervised learning systems [Kan03]. Association rule mining is initially used for the market basket problem. For example in a market basket database, in 80% of the cases when people buy *bread*, they also buy *milk*. This tells us that item *bread* is closely associated with item *milk*.

Generally, an association rule is an implication $X \Rightarrow Y$ occurring in transactions of a database (or dataset), where X and Y are sets of items or *itemsets* [AS94]. Given a database, there would be a huge number of possible rules. The problem of association rule mining is to search for only "interesting" rules. There are two issues in association rule mining. The first issue is how to determine whether a rule is interesting, and the second one is how to search for all interesting rules from the huge space of possible ones.

Conventionally, an association rule is considered mainly based on the support and confidence measures. While *support* is seen as usefulness - how frequent the rule is observed, *confidence* measures the implication relationship - how certain the rule is [HK01]. An association rule is considered to be interesting if the values of support and confidence are greater than or equal to predefined support and confidence thresholds respectively. However, the support-confidence framework has a major drawback in which the number of rules that are mined under this framework is too large and many of them are uninteresting [Heg03]. This may be due to the following two reasons:

- *The support and confidence measures are not sufficient enough to cover all characteristics of association rules.* Other measures such as *interest*, *conviction*, *reliability* and *certainty factor* [BRS97, BMUT97, AEMT00, BBSV02] have been proposed in order to further characterize association rules. Apart from determining the co-occurrence and implication of items as support and confidence measures, the other measures also present the association of item distributions in the transactions of the database.

- *Users may be interested only in certain types of data items.* For example, garment producers may only deem a rule to be interesting if it contains items related to garment production. This kind of measure is considered as *constraint* for association rules [SVA97, NLHP98]. There are two possible values for a constraint - *true* for interesting rules, and *false* for uninteresting ones. Note that association rules mined must satisfy at least those constraints on *support* and *confidence*, which stipulate that the support and confidence values of the rules must be greater than or equal to the support and confidence thresholds respectively.

Searching for all interesting rules from the large space of possible rules is a challenge in association rule mining. Given a database of m items, the number of possible itemsets is 2^m. For an average number of items of a typical market basket of 100, there are $2^{100} \approx 10^{30}$ possible itemsets. Therefore, scanning through all the possible rules for interesting ones is almost impossible.

To reduce the combinatorial search space, most of the association rule mining algorithms are based on the anti-monotone property [AS94] of the support constraint on the set of items of rules. The anti-monotone property states that if a rule does not satisfy the support constraint, then any other rules which contain all items of that rule will not satisfy the support constraint either. As such, we can prune those rules from the search space without counting the support value. Another approach to tackle the huge search space problem is to integrate constraints into the rule mining process [SVA97, NLHP98]. If a rule does not satisfy the mining constraints, then it can be pruned without counting the support and confidence values. Thus, the constraint can be used for limiting the search space and thereby reducing the complexity of the process of mining association rules.

Although many approaches have been proposed for association rule mining [AS94, HP00, SVA97], there remain computational problems related to generation of an overwhelmingly large number of association rules [Heg03]. Therefore, appropriate rule assessment and constraint mining techniques need to be investigated to reduce the complexity of the mining process as well as to process the mined set of rules effectively.

1.2 Associative Classification

Besides the original role of showing item relations in market basket problems, association rules can also be used for prediction. For example, an association rule $R : X \Rightarrow Y$ can be used to predict that if a transaction contains a set of items X, then it will probably contain

a set of items Y. When itemset Y is limited to a data item in the set of class labels (i.e., Y is a class label), the prediction using the association rule R is basically a classification task. The association rules that can be used to predict class labels are called *class association rules* (CARs).

The support measure gives the frequency of a rule that has occurred in a dataset. A rule with a high support value means that it occurs frequently in the dataset. The rule can then be considered as a general pattern that can be used to predict missing data or summarize the dataset. The confidence measure presents the certainty of the "if X then Y" implication, in which one may predict that a data sample belongs to class Y if it contains itemset X. As such, the support and confidence constraints ensure both the usefulness and certainty of class association rules. This guarantees high accuracy of the classification process that uses these class association rules.

The integration of association rule mining and classification was first introduced in [LHM98] via the Classification Based on Associations (CBA) algorithm. A set of class association rules used for associative classification can be generated using conventional association rule mining algorithms such as the Apriori algorithm [AS94]. However, the integration of association rule mining and classification, the two very different data mining tasks, is not straightforward. To mine an effective set of class association rules for AC, we need to consider the followings:

- *The complexity of the rule mining process.* The mining process returns all association rules satisfying the support and confidence constraints. This is important for the classification process as the information-rich set of rules will enable it to achieve optimal prediction in classification. However, gathering all association rules by a conventional association rule mining process also entails exhaustive and very computationally expensive search in a huge space of possible rules.

- *The trade-off between rule mining complexity and classification accuracy.* When the support and confidence constraints are tight (i.e., support and confidence threshold values are set high), the mining process is fast and returns a small number of rules. However, this small set of rules usually leads to an inaccurate associative classification. When the support and confidence constraints are relaxed (i.e., support and confidence threshold values are set lower), more rules can be obtained. This help achieve a more accurate classification. However, the relaxation of the support and confidence constraints will also increase the complexity of the mining process. This is especially significant for large datasets.

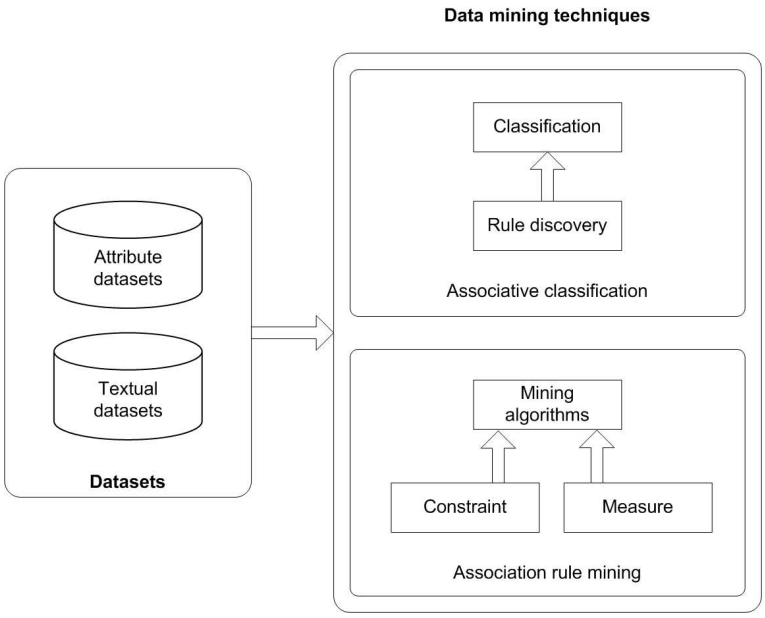


Figure 1.1: The proposed research framework.

1.3 Objectives

This research aims to develop effective techniques for enhancing association rule mining and associative classification. In particular, for association rule mining, we investigate new techniques for measure and constraint for association rules, and also associated rule mining algorithms for mining interesting rules. For associative classification, we investigate new techniques for discovering association rules effectively for classification, especially from large datasets. Figure 1.1 presents the proposed research framework.

1.4 Contributions

As a result of this research, we have developed different effective techniques for association rule mining and associative classification. The contributions of this research are listed as follows:

- A new type of constraint for association rules called *category-based constraint* is proposed. In addition, an Apriori-based algorithm called Category-based algorithm (Apriori^{CB}) is also proposed for mining association rules with the category-based constraint. By using category patterns, the Apriori^{CB} algorithm is faster and requires much less memory compared with other association rule mining algorithms. This advantage is

especially significant when mining a large number of frequent itemsets and frequent itemsets containing many items.

- Effective measures for assessing association rules are analyzed. Besides support and confidence, the relative confidence of implication relationship, which is captured by the certainty factor (CF) measure [SB90], is able to reflect truthfully the implication relations of association rules. This can be used to overcome drawbacks on generating many misleading rules in the conventional support and confidence framework .

- A new approach called *associative feature selection* is proposed for unsupervised feature selection from textual datasets. The proposed approach is based on associative relations among items, which are represented by measures such as confidence and CF, to compute relevant features. The proposed associative feature selection approach can be used for data preprocessing of textual datasets, which are generally large and contain a lot of irrelevant features.

- A multiple-step rule discovery approach for associative classification called *Mstep-AC* is proposed. The proposed approach uses a multiple rule mining and rule combination mechanism to gather a set of rules for AC. This mechanism is able to discover more rules for AC with the same support threshold and to assess the discovered rules with more appropriate ranges of data in comparison with conventional AC approaches. According to our performance evaluation, the proposed approach has achieved better associative classification accuracy while maintaining the rule discovery process at a complexity level comparable with conventional AC approaches.

- An approach called *AIS-AC*, which is based on the Artificial Immune System (AIS) [dCVZ99, dCT03, DZ03], is proposed. Instead of massively searching for all possible association rules, the AIS-AC approach only finds a subset of association rules for associative classification in an evolutionary manner. The proposed approach aims to reduce the computational complexity while maintaining the same level of accuracy. As a result, the proposed approach can work with very small values of the support threshold, which normally cannot be managed in conventional approaches. According to our performance evaluation, the proposed AIS-AC approach can discover effective association rules for AC from large datasets.

1.5 Organization of the Thesis

This chapter has briefly introduced some background on association rule mining and associative classification, and stated the motivation for this research. The objectives and contributions are also discussed. The rest of the thesis is organized as follows.

Chapter 2 reviews the related work on association rule mining, associative classification and association rules for text mining.

Chapter 3 discusses the proposed category-based constraint and the Apriori^{CB} mining algorithm. The performance evaluation of the proposed Apriori^{CB} algorithm is also given.

Chapter 4 discusses the effective measures for assessing association rules. In addition, the proposed associative feature selection approach for text mining and its performance are also covered by this chapter.

Chapter 5 discusses the proposed Mstep-AC approach for mining class association rules for associative classification. The performance evaluation of the proposed Mstep-AC approach is also given.

Chapter 6 discusses the proposed AIS-AC approach for mining association rules for associative classification. The performance evaluation of the proposed AIS-AC approach is also given.

Finally, Chapter 7 concludes this thesis and proposes some new directions for future research.

Chapter 2

Related Work

In this chapter, we review the basic concepts and techniques in association rule mining and associative classification. We also discuss association rules for text mining. This chapter is organized as follows. First, we review the concepts of association rule mining and its related work. Next, we review the concepts relevant to associative classification and its related work. Then, we review association rule mining and associative classification for text mining. Finally, a summary of the chapter is given.

2.1 Association Rule Mining

Association rules [AIS93, AS94] tell us about the relations between two or more items. For example in a market basket problem, in 80% of the cases when people buy bread, they also buy milk. This tells us that item bread is closely related to item milk. This relation is then presented as a "if bread then milk" rule with a *confidence* of 80%. The *support* of the rule is the probability of people buying both bread and milk. It guarantees that the rule is frequently observed. According to [HK01], while the support measure presents the usefulness of the rule, the confidence measure presents the certainty of the rule. The goal of association rule mining is to find all the useful and certain association rules that have the support and confidence greater than or equal to given threshold values.

2.1.1 Concepts

Let $I = \{i_1, ..., i_m\}$ be a set of m distinct literals called *items*. Let *database* (or *dataset*) D be a collection of transactions where each transaction T is a set of items in I (i.e., $T \subseteq I$).

An itemset S is a set of items in I (i.e., $S \subseteq I$). An itemset S is contained in transaction T if and only if T contains every item of S (i.e., $S \subseteq T$). A *k-itemset* is an itemset of k items.

An *association rule* R is an implication $X \Rightarrow Y$, where X and Y are sets of items in D and $X \cap Y = \emptyset$. The confidence of the rule $conf(R)$ is the percentage of transactions in D that contains Y among the transactions containing X. The support of the rule $supp(R)$ is the percentage of transactions in D containing both X and Y with respect to the total number of transactions in D [AS94]. Itemset X is called the *antecedence* of rule R, and itemset Y is called *consequence* of rule R.

Let $P[S]$ be the probability of itemset S occurring in transactions of D. Then, $P[S]$ is the percentage of transactions containing S with respect to the total number of transactions in D. Similar to the definition of support of a rule, $P[S]$ can be considered as the support of itemset S (denoted as $supp(S)$). We call $P[X]$ (or $supp(X)$) the *antecedence support* of rule R, and $P[Y]$ (or $supp(Y)$) *the consequence support* of rule R.

Assume that the database contains N transactions with the numbers of transactions that contain X, and both X and Y are a, and b respectively. It can be implied from the definitions of support and confidence of association rules that $supp(R) = b/N$ and $conf(R) = b/a$; and from the notation of support of itemsets that $P[X] = a/N$ and $P[X \wedge Y] = b/N$. We can calculate the values of $supp(R)$ and $conf(R)$ using $P[X \wedge Y]$ and $P[Y]$ as follows [Ada01]:

$$supp(R) = P[X \wedge Y] \tag{2.1}$$

$$conf(R) = \frac{P[X \wedge Y]}{P[X]} \tag{2.2}$$

Mathematically, association rule mining means finding all the association rules R such that $supp(R)$ and $conf(R)$ are greater than or equal to predefined support and confidence thresholds *minSupport* and *minConfidence* respectively as follows [AS94]:

$$supp(R) \geq minSupport \tag{2.3}$$

$$conf(R) \geq minConfidence \tag{2.4}$$

The association rules satisfying conditions (2.3) and (2.4) are called *strong* rules [HK01]. In other words, association rule mining means finding all strong rules from a given database. Let $Z = X \cup Y$, then $supp(R) = P[X \wedge Y] = P[Z]$. Therefore, rule R satisfying condition (2.3) means that the support of itemset Z is greater than or equal to *minSupport* (i.e., $P[Z] \geq minSupport$). Itemset Z is then called a *frequent* itemset.

One of the possible ways of mining association rules is to first look for all frequent itemsets Z, then test rules of possible pairs of itemsets X and Y such that $Z = X \cup Y$ and $X \cap Y = \emptyset$. As Z is a frequent itemset, the condition (2.3) for the support of R holds. Therefore, rule R is strong if condition (2.4) is also satisfied. Most of the association rule mining algorithms follow this framework [AS94, Ada01] for searching association rules. The above two-step procedure is summarized below:

- Find all frequent itemsets Z from database D such that

$$supp[Z] \geq minSupport \tag{2.5}$$

- For each itemset Z found in Step 1, generate association rules $R : X \Rightarrow Y$ such that

$$(X \subseteq Z) \wedge (Y = Z \setminus X) \wedge (conf(R) \geq minConfidence) \tag{2.6}$$

The second step is straightforward. For each itemset Z, X can be any sub-itemsets of Z. For each X, the confidence value $conf(R)$ of rule R can be computed based on $P[X]$ and $P[X \wedge Y] = P[Z]$ as shown in formula (2.2). If $conf(R) \geq minConfidence$ (condition (2.4) holds), then R is a strong rule with consequence Y calculated as $Y = Z \setminus X$. Note that $(X \subseteq Z) \wedge (Y = Z \setminus X)$ is equivalent to $(Z = X \cup Y) \wedge (X \cap Y = \emptyset)$.

Therefore, the main task of association rule mining is to discover frequent itemsets. This is generally a very costly process. Given a set of items I size m, the number of distinct subsets is 2^m. For an average number of items of a typical market basket, say, up to 100, there are $2^{100} \approx 10^{30}$ subsets. Clearly, visiting all possible subsets to find frequent ones is very time consuming. To reduce the combinatorial search space, most association rule mining algorithms are based on the property of support that "any subsets of a frequent itemset are frequent" [AS94, Heg03]. In other words, all supersets of an infrequent itemset must be infrequent. Many algorithms have been proposed for effectively mining frequent itemsets using this property of support. If an itemset is not frequent, then all of its supersets can be pruned from the search space without counting the support values [AS94, Heg03, Goe04].

The search strategy for frequent itemsets can be divided into two types: breadth-first and depth-first. In the *breadth-first* (also called level-wise) search strategy [AS94, SON95], the search is moved from itemsets with the least number of items to itemsets with the most number of items. The process of breadth-first search algorithms usually includes n passes where n is the maximal number of items in an itemset. In the first pass, all 1-itemsets

(itemsets of a single item) are considered. In pass k ($k > 1$), itemsets containing less than k items, which are obtained from the previous $(k-1)$ passes, are used to explore k-itemsets.

The *depth-first* search strategy [ZPOL97, HP00, HPY00, SHS+00] assumes certain ordering on the set of items. The search is carried out from itemsets with items of high orders to itemsets with items of low orders. This process usually includes m passes where m is the number of items. In the first pass, the set of itemsets consists of one itemset formed from the top item. In pass k ($k > 1$), the itemsets containing the top $(k-1)$ items, which are obtained from the earlier $(k-1)$ passes, are used to explore the itemsets containing the top k items.

In the next few subsections, we review some of the popular association rule mining algorithms, namely Apriori, Partition, Eclat and FP-growth. Due to the simplicity of the second step of the two-step procedure in association rule mining, our discussion of related association rule mining algorithms focuses on the searching process for frequent itemsets.

2.1.2 Apriori Algorithm

Agrawal and Srikant [AIS93, AS94] proposed the *Apriori algorithm*, which is the best known association rule mining algorithm using the breadth-first search strategy. The Apriori algorithm has sparked the development of various kinds of Apriori-based algorithms [LHM98, NLHP98, SON95, MTV93]. It searches for frequent itemsets using a *generate and count* framework. In this framework, only *candidate itemsets*, which have frequent subsets, are generated and then counted for their supports. In the first pass, it counts the supports for all 1-itemsets and gathers frequent 1-itemsets L_1. In pass k ($k \geq 2$), the candidate k-itemsets C_k, which are generated from the frequent $(k-1)$-itemsets L_{k-1} collected from the last pass, are counted to gather frequent k-itemsets L_k. The property of support guarantees that candidate k-itemsets L_k cover all frequent k-itemsets C_k. In other words, this approach does not miss any frequent itemsets while narrowing down the counting space.

In each pass, the counting for the support of candidates takes the most runtime as the counting operation involves processing of all transactions in the database for each candidate. For fast support counting, candidate itemsets are partitioned into nodes and stored in a *hash-tree* structure [AS94]. During the support counting, each transaction is hashed into its appropriate nodes in the hash-tree. As such, instead of comparing each candidate to every transaction in the database, the support count of the candidates can be obtained by gathering the number of transactions that are hashed into their corresponding nodes [HK01].

The advantage of the Apriori algorithm is that using the breadth-first search strategy, an itemset is explored after all of its subsets. That is, before counting the support of a candidate itemset, it is possible to check all subsets of it. The candidate itemset is pruned if any of its subsets are not frequent. We may say that the algorithm fully exploits the property of support to help minimize the number of candidate itemsets, and thereby minimize the support counting. However, the drawback of the Apriori algorithm is that it requires multiple passes for determining the set of frequent itemsets. In each pass, it requires scanning the database once, generating candidate itemsets and hashing transactions into the hash-tree of candidate itemsets.

2.1.3 Partition Algorithm

Savasere *et al.* proposed the *Partition algorithm* [SON95] that divides the database into a number of non-overlapping partitions. The partition size is chosen so that each partition can be fitted into the main memory. The Partition algorithm is carried out in two steps. In the first step, it generates frequent itemsets for each partition. All these locally generated frequent itemsets are then combined together to form the set of candidate itemsets. In the second step, actual supports of these candidate itemsets are counted and frequent itemsets in the whole database are then identified. Hence, the rationale is that if an itemset is globally frequent in the database, it must be locally frequent in at least one partition. That is, the set of frequent itemsets collected from all partitions contains all actual frequent itemsets.

Similar to the Apriori algorithm, the process of generating frequent itemsets for each partition is carried out in the breadth-first search manner. In pass k, the frequent $(k-1)$-itemsets obtained from the previous passes are used to generate the candidate k-itemsets which are then counted for the support to determine whether they are frequent k-itemsets. However, the operation of scanning the database in each pass is not necessary as the partitions are stored in the main memory. Thus, the Partition algorithm only scans the database once for generation of locally frequent itemsets in the first step. However, it may need to count many itemsets for the support which are locally frequent in a certain partition but not globally frequent in the database. In addition, it also requires an extra amount of system memory to store the database partitions in comparison with the Apriori algorithm.

2.1.4 Eclat Algorithm

Zaki *et al.* proposed the *Eclat algorithm* [ZPOL97, Zak00] which uses a vertical data layout format to search for frequent itemsets. In the vertical data layout format, a record consists of an item and its *tid-list*, which contains all identifications (denoted as *tid*) of the transactions in the conventional horizontal database that contains this item. Similarly, the tid-list of an itemset S (denoted as $tid\text{-}list(S)$) contains all tids of the transactions that contain this itemset. The support of an itemset X can then be obtained by simply intersecting the tid-lists of some subsets of it [ZPOL97].

The Eclat algorithm employs the depth-first search strategy for frequent itemset searching. Let's denote the sequence of ordered items as $(i_1, ..., i_m)$ and F_l as the set of frequent itemsets containing the top l items. In pass k, the candidate itemsets can be obtained by adding i_k into frequent itemsets F_{k-1} obtained from the previous $(k-1)$ passes. That is, a candidate itemset has the form of $S' = S \cup \{i_k\}$, where S is a frequent itemset in F_{k-1}. The tid-list of S' can be obtained by intersecting the tid-list of S and the tid-list of item i_k such that $tid\text{-}list(S') = tid\text{-}list(S) \cap tid\text{-}list(i_k)$. For fast intersection, Eclat uses $tid\text{-}list(\{i_k, i_j\})$ instead of $tid\text{-}list(i_k)$ where i_j is an item in S. As i_j is in S, $S' = S \cup \{i_k\} = S \cup \{i_k, i_j\}$, or $tid\text{-}list(S') = tid\text{-}list(S) \cap tid\text{-}list(\{i_k, i_j\})$. Generating the support of S' using the tid-list of $\{i_k, i_j\}$ is generally much faster than using the tid-list of i_k as the number of transactions containing both i_k and i_j is possibly much less than the number of transactions containing i_k. Eclat performs database preprocessing to gather all tid-lists of frequent 2-itemsets. Thus, $tid\text{-}list(\{i_k, i_j\})$ is available for the intersection operation.

The advantage of Eclat is its more efficient support counting compared with the laborious iterative support counting mechanism of the breadth-first search used in Apriori-based algorithms [Goe04]. However, the Eclat algorithm requires extra amount of system memory for storing tid-lists of frequent itemsets for fast intersection. Although some techniques [ZG03] are available for reducing the amount of required memory, the Eclat algorithm still requires the original database to be stored in the main memory [Goe04]. In addition, the Eclat algorithm does not fully exploit the property of support as candidate itemsets are generated based on only two of its subsets. Thus, the number of candidate itemsets is much larger in comparison with the breadth-first search used in Apriori-based algorithms [Goe02].

2.1.5 FP-growth Algorithm

Han *et al.* proposed the *FP-growth algorithm* [HP00, HPY00] in which *Frequent Pattern Tree* (FP-tree) is used to store the database in a combination of the horizontal and vertical data layout formats. The FP-tree is a *trie* structure [AFK97, Sed88] with a counter in each node to keep track of the number of transactions that share the branch through that node. There is also a *node-link* in each node that links to the next node in the FP-tree carrying the same item. In addition, there is a header table in which each entry corresponds with an item and contains a link to the first node in the FP-tree carrying the same item. Here, every branch starting from the root node represents a compressed form of the transaction in the horizontal data layout format, whereas every linked-list starting from the header table represents the compressed form of tid-lists of items in the vertical data layout format [Goe02].

In FP-growth, the process of mining frequent itemsets is carried out using the depth-first search strategy similar to that of Eclat. The support of a candidate can be computed from the corresponding tid-lists obtained by following the linked-lists starting from the header table. The FP-tree with the compressed form of tid-lists leads to a faster support counting operation in comparison with that of Eclat. When the FP-tree can fit into the memory, the algorithm is efficient for the discovery of long-patterns. However, the drawback of the FP-growth algorithm is the complex data structure of the FP-tree, in which every node needs to store an item label, a counter, and three pointers to the next node, the node's branch and its parents. Thus, the actual size of FP-tree is often much larger than the plain array-based structure of Eclat [Goe04].

2.1.6 Discussion

Hipp *et al.* [HGN00] compared the popular association rule mining approaches including Apriori, Partition and Eclat. These approaches have shown similar runtime behaviors and no algorithm fundamentally outperforms the others. For example, Apriori is superior in mining the market basket database, but it performs poorly with the car equipment database. The FP-growth algorithm is very efficient in many cases, but it requires a large amount of memory to store the FP-tree [Goe04]. Although there may be differences with different implementations and databases, association rule mining algorithms exhibit the same performance behavior with respect to the support threshold value. For example, when the support threshold is decreased, the number of frequent itemsets increases exponentially [AS94, HPY00, HGN00].

One of the problems of association rule mining is the computational complexity, which is related to the large number of frequent itemsets and the size of the frequent itemsets themselves [Heg03]. In the Apriori algorithm, for every frequent itemset, we need to find the supports of all its subsets before determining whether the itemset is a candidate. Thus, for a k-itemset, the support of all 2^k subsets will have to be determined. It means that with a 20-itemset, there are more than one million subsets need to be processed. In addition, many interesting rules have a high confidence but a low support value [Heg03]. That is, to ensure that interesting rules will not be missed, the support threshold needs to be decreased. This will probably result in an exponential increase in runtime and resource usage (i.e., memory and disk space) during the frequent itemset mining process.

2.2 Associative Classification

Classification is learning a function that can map or classify data into one of several predefined classes [WK91]. The learned function is also referred to as *classification model* or *classifier*. A classification model is constructed by analyzing data samples from a *training dataset*. Since the class of each training sample is provided, the construction process of a classifier is a *supervised learning* process, in which the class of each training sample is labelled [HK01].

Associative classification (AC) [LHM98, LMW00] takes advantage of association rule mining [AS94, HPY00] in the rule discovery process in extracting high quality rules that can accurately generalize the training dataset for data classification. This approach can achieve high accuracy in comparison with other classification approaches [LHM98, LHP01, LMW01, AZ02]. In associative classification, the implication relation of an association rule which is discovered from the training dataset is used to predict the class labels of new data samples. If a data sample contains the antecedence of a rule, the consequence of the rule will be used to predict the class label of the data sample.

2.2.1 Concepts

In AC, association rules are searched from the training dataset in the form of $iset \Rightarrow c$, where $iset$ is an itemset and c is a class. These association rules are called *class association rules* (CARs) [LHM98]. The implication of a class association rule $R : iset \Rightarrow c$ means that if a data sample contains $iset$, then it probably belongs to class c. The data sample that contains $iset$ is said to have matched rule R. Thus, if a data sample matches rule R, it will be classified into class c of R.

The support measure gives the frequency of a rule. A rule with a high support value means that it occurs frequently in the dataset. The rule can then be considered as a general pattern that can be used for data classification. On the other hand, if a rule has a small support value, it may occur only in few transactions. Such rules may show specific patterns pertaining only to a small number of transactions, or they may simply be noise in the dataset. Therefore, these rules tend to overfit the data and are generally not useful in representing any knowledge from the dataset. These rules can be eliminated in the rule mining process using the support threshold.

The confidence of a rule $R : iset \Rightarrow c$, say 80%, guarantees the reliability of the classification decision. The test sample is classified into class c as most of the transactions (i.e., 80%) that contain itemset X fall into class c. The confidence measure gives the certainty of the implication "if $iset$ then c", in which one may predict that a data sample belongs to class c if it contains itemset X. In other words, the accuracy of prediction is determined by the confidence measure. This means that only rules with the confidence values above an threshold are useful. And rules with a low confidence value should be eliminated using the confidence threshold.

Therefore, the support and confidence constraints are necessary for identifying CARs for accurate classification prediction. In other words, it is possible to discover all accurate CARs using the association rule mining framework, in which all rules satisfying the support and confidence constraints are obtained. Generally, AC algorithms consist of three major processes, namely rule discovery, rule selection and classification [LHM98], as follows:

- *Rule discovery.* This process discovers a complete set of CARs from a training dataset using given support and confidence threshold values.
- *Rule selection.* This process evaluates the rules derived from the rule discovery process and selects a subset of them for data classification.
- *Classification.* This process classifies new data samples based on the set of rules derived from the rule selection process.

The rule discovery process adopts association rule mining approaches for discovering a set of CARs that satisfy the support and confidence constraints. The rule discovery process is complex as it requires exhaustive search in the huge space of possible CARs. In addition, the process may return a large number of discovered rules which may lead to a complex rule selection process [YH03]. Due to the interactions among rules, not all of the discovered rules

are accurate when they are used in the classification process. Thus, it is necessary to filter out some of the discovered rules through the rule selection process in order to achieve accurate classification. As there are a large number of possible subsets of rules that can be selected from the discovered rules, most approaches select a set of rules heuristically [LHM98, LHP01]. In the classification process, to classify a data sample, the process needs to decide which rules are chosen for classifying data samples from a set of selected rules and how to derive the class label from chosen rules [LHM98, LHP01].

2.2.2 Classification Based on Associations

Associative classification was first introduced in [LHM98] by Liu *et al.* with the *Classification Based on Associations* (CBA) algorithm. The algorithm proposes the general AC framework which is used by most of the subsequent AC algorithms [LMW00, LHP01, LMW01, YH03]. In the rule discovery process, CBA adopts the Apriori algorithm to discover the complete set of CARs that satisfy the support and confidence constraints. It differs from Apriori in that it searches for frequent *ruleitems*, each of which consists of an itemset and a class, instead of frequent itemsets [LHM98]. Each frequent ruleitem can then be used to form a CAR that satisfies the support constraint. From the CARs generated using the discovered frequent ruleitems, set of CARs satisfying the confidence constraint is then extracted.

In the rule discovery process, discovered rules are first ranked using the support and confidence values, and the number of items of the rule [LHM98, LHP01]. The ranking of CARs relies mostly on the confidence measure with the assumption that rules with higher confidence would probably give better classification accuracy. Next, a subset of rules is selected by scanning discovered rules in descending order. For each discovered rule, if it can correctly classify at least one data sample, then it is selected and the data samples that can match the rule are removed from the training dataset [LHM98]. When the training dataset becomes empty, the set of selected rules will cover the whole training dataset such that, for each training data sample, there is at least one selected rule matching it. To classify a data sample d, CBA simply takes the rule R with the highest ranking that matches d. The test sample d is then assigned to class c of R [LHM98].

The strength of CBA is that, based on association rule mining, it can discover all accurate rules that satisfy the support and confidence constraints. In other words, it can search exhaustively for all accurate CARs and thereby achieving global optimality [Yan03]. This is different from conventional classification algorithms such as C4.5 [Qui93] and rule induction

[Coh95], in which greedy search is used to build the classifiers. Using exhaustive search, the CBA algorithm may achieve high accuracy. However, the rule searching process is complex in comparison with conventional classification algorithms [LHM98, YH03].

To avoid performance degradation of the rule discovery process, a maximum number of candidate rules (e.g., 80,000) has been set in the CBA algorithm [LHM98]. However, this limitation may lead to a resultant set of rules that contains only a few items. For example, let's consider a dataset which contains only 2 classes and 300 items. Assume that the support threshold is set to be small such that all 300 items are frequent. The number of candidate rules containing two items is then $C^2_{300} \approx 45{,}000$ for each class. It means that the number of candidate rules containing two items of both classes is around 90,000. Thus, if the maximum number of 80,000 candidate rules is set, the discovery process will return a set of rules containing only one or two items. Therefore, in this case, the maximum number of candidate rules must be increased substantially in order to discover a sufficient set of rules for accurate AC. However, this will lead to a more complex rule discovery process. In other words, the problem on the complexity of the rule discovery process for accurate AC still remains.

2.2.3 Classification based on Associations with Multiple Minimum Class Supports

For real-world datasets, class frequency may be distributed very unevenly. That is, there may be frequent classes which occur in many data samples and infrequent classes which occur in only a few data samples. When a single support threshold is used, we may not find sufficient rules of infrequent classes, but obtaining too many useless and overfitting rules for frequent classes [LMW01].

In the *Classification Based on Associations with Multiple Minimum Class Supports* (msCBA) algorithm [LMW00, LMW01], Liu *et al.* suggested the use of *multiple minimum class supports* to produce a more balanced number of association rules for different classes during the rule discovery process. The support threshold for a class c is calculated as a product of the *total minimum support threshold* given by users and the *frequency distribution* of the class c. The frequency distribution of the class c is the percentage of the number of data samples containing c with respect to the total number of data samples. Apart from the support and confidence threshold values, msCBA is similar to CBA in all three processes of AC. The use of multiple minimum class supports helps obtain a balance among the rules

of different classes. This helps msCBA achieve a more balanced classification performance among different classes in comparison with CBA.

2.2.4 Classification based on Multiple Association Rules

In [LHP01], Li *et al.* proposed the *Classification based on Multiple Association Rules* (CMAR) algorithm which adopts the FP-growth approach for discovering CARs in the rule discovery process. The rule selection process is similar to CBA in rule ranking. However, the difference is that there is an extra step in which non-positive correlated rules are removed. The correlation is measured by using the chi-square test statistics (χ^2) [LHP01]. Moreover, during the rule selection process, a data sample is removed from the training dataset when there are δ selected rules matching it. Thus, when the training dataset becomes empty, the set of selected rules covers the training dataset with the depth of δ such that, for each training data sample, there is at least δ selected rules matching it. In the classification process, multiple rules are used instead of a single rule for classification of a data sample. That is, all rules that matched a test sample are collected and grouped according to their classes in order to classify the test sample. The test sample is then assigned to the class from the best group based on a statistical measure called $max\chi^2$.

CMAR employs the FP-growth approach for searching a set of CARs. Thus, it requires the dataset to be stored in the main memory in the compact form of FP-tree and searches for the set of CARs in the depth-first manner. The multiple database coverage of rules for a data sample in CMAR may generate more rules for the classification process which also uses multiple rules for classifying a data sample. The selection and classification processes of CMAR are, therefore, more complicated in comparison with AC approaches that use single database coverage and single rule classification such as CBA and msCBA.

2.2.5 Classification based on Predictive Association Rules

Different from most AC approaches, an approach called *Classification based on Predictive Association Rule* (CPAR) [YH03] does not follow the association rule mining framework for rule discovery for AC. The algorithm applies FOIL [QCJ93], a first-order learning system, to generate a set of rules for classification. A rule is learned by repeatedly adding literals (i.e., items) that give the best information gain (called FoilGain). CPAR modifies FOIL by carrying out multiple rule searching processes simultaneously. It keeps not only the best items but also all close-to-the-best items [YH03]. This approach, therefore, gathers a larger set of

rules in comparison with FOIL. The rule searching process is not based on association rule evaluation measures such as support and confidence. CPAR may be seen as a conventional rule-based classification, but a data sample may match multiple rules. This approach is similar to conventional AC approaches only in deriving class labels from multiple rules during the classification process.

2.2.6 Discussion

The CBA and msCBA algorithms have utilized the breadth-first search based Apriori approach for discovering the set of CARs in the rule discovery process. In contrast, the CMAR algorithm adopts the FP-growth approach with the FP-tree data structure in its rule discovery process. Thus, there are differences in performance in terms of runtime and memory usage. CMAR with the FP-growth approach may achieve efficient rule discovery process for some training datasets. However, like the FP-growth algorithm, it requires an extra amount of memory to store the FP-tree [Goe04].

Another consideration in the rule discovery process is the selection of appropriate values for the support and confidence thresholds. In [LHM98], it is reported that the rule discovery process can use the support threshold in the range of [1%-2%] and the confidence threshold of 50% as these values give the best average accuracy based on 26 datasets it has used for performance evaluation. Many other AC approaches [LHP01, LMW01, YH03] have also applied these values of the support and confidence thresholds for the rule discovery process. However, the optimal values of the support and confidence thresholds depend on individual datasets. For example, Baralis *et al.* [BCG04] pointed out that the optimal accuracy has been achieved with very different support threshold values for those 26 datasets, and the value of the support threshold of [1%- 2%] is far from optimal for some datasets.

The appropriate support value for AC also relates to the size of the dataset. Generally, the support threshold value gives the statistical significance of the class association rules [CB91]. It ensures that the accuracy of the rule observed in the training dataset, which is the confidence of the rule, is close to the rule's actual prediction accuracy. However, this actually depends on the number of observed samples (i.e., the number of data samples containing the itemset of the rule). Thus, it is important to consider the support value based on the number of data samples rather than the absolute percentage such as 1% or 2%. For example, let's consider two training datasets of 1,000 and 10,000 samples, respectively. A rule discovered from the first dataset with support of 1% (i.e., it occurs in 10 data samples) should be less

accurate than that of another rule discovered from the second dataset with the same support and confidence values which occurs in 100 data samples. In the second dataset, even a rule with a small support value of 0.5% will occur in 50 data samples, which are statistically more reliable for estimating the prediction accuracy from the confidence value. Therefore, infrequent rules (i.e., with small support values) can be used in large datasets to achieve accurate prediction. In other words, to achieve high accuracy in associative classification from large datasets, the rule discovery process may be very computationally expensive.

In the classification process, the class of new data samples can be derived using a single rule or multiple rules. It is claimed that the multiple rule approach is better as it combines the implication of all rules matching the data sample [LHP01]. However, the interaction and dependency among rules may affect the classification performance using multiple rules. For many datasets, single rule classification approaches may achieve better accuracy compared with multiple classification approaches [Li01].

According to the performance results given in [LHM98] based on 26 selected datasets from UCI Machine Learning Repository [NHBM98], CBA has achieved an average accuracy of 84.5%, which is better then C4.5 [Qui93], which has an average accuracy of 83.3%. Using the same 26 datasets, msCBA and CMAR have achieved an average accuracy of 85.1% and 85.2% respectively, showing further improvement over the CBA algorithm. When comparing msCBA and CMAR, the two algorithms have achieved similar accuracy, but the msCBA algorithm has simpler implementations in both rule selection and classification.

2.3 Association Rules for Text Mining

Apart from transactional databases, association rule mining, as a general data mining technique, can also be applied to textual databases which are usually organized as a collection of textual *documents*. A document can simply be considered as a transaction of *words* (or *phrases*) [RB98] in a typical *bag-of-word* representation [Seb02]. However, such a representation has problems in that the same word may be expressed entirely with different meanings (e.g., polysemous words) or different words may have the same meaning (e.g., synonyms) [NN01]. On the other hand, a document can also be considered as a transaction of *concepts* that occur in the document [RB98, FH97] using *semantic concept* representation, which can help deal with the problems caused by polysemy and synonymity [NN01].

2.3.1 Association Rule Mining for Text Mining

Before mining association rules from a textual database, one first needs to convert raw textual documents into the standard format of transactions of words in the bag-of-word representation, or concepts in the semantic concept representation. For the bag-of-word representation, data preprocessing can be done automatically and usually comprises the following steps [AE99]. Firstly, it removes useless information, such as punctuation marks and HTML tags, from raw textual documents. Secondly, it removes stopwords which are frequent words, such as pronouns, prepositions, etc. Thirdly, it performs word stemming by removing word suffix for grouped words that denote the same conceptual meaning, such as walk, walked and walking [AE99]. For the semantic concept representation, the preprocessing process is much more complicated as it needs to extract concepts from the textual document collection. Concept extraction is a difficult computational linguistic task. It may require manual efforts from experts and therefore can be laborious and time consuming.

After preprocessing textual documents into the standard format of transactions of words or concepts, the process of association rule mining from a textual database is basically similar to that of a transactional database. However, textual transactional databases have some features that are considerably different from that of conventional transactional databases. Firstly, it may consist of a very large amount of textual documents. This means that the discovery process is complex, requiring long runtime and large amount of computing resources. Secondly, a transaction representing a document may contain hundreds or thousands of items (words), and many of them tend to occur together. This may also lead to a complex rule discovery process with a large set of discovered rules. Thirdly, the item frequency distribution can be very uneven. Some frequent words may occur in many documents while some infrequent ones may occur in only few documents. Consequently, the set of discovered rules can also be very uneven with many rules of frequent items but only a few rules of infrequent items.

To deal with the problems concerning huge databases and long transactions, most current approaches simply filter out ambiguous and unimportant items (words). This can be done manually by experts [CT02] or automatically by using feature selection approaches [LMY03]. To deal with a large number of discovered rules, one possible approach is to use "rule templates" or constraints which specify the type of rules to be mined in order to filter

out unwanted rules [KMR+94]. In addition, other measures (besides the support and confidence measures) for association rules that reflect complex semantic relations among items can also be used to effectively gather interesting rules [CT02].

Association rule mining has been applied to textual databases in different applications. When a document collection is represented using bag-of-words, association rules show the relations among words. This kind of relations can be used for automatic linguistic processing such as determining word compounds [RB98] or structuring terminologies [CT02]. Moreover, the relations among words presented using association rules can also be used for extracting concepts from a collection of textual documents [MS00]. On the other hand, when a document collection is represented using semantic concepts, association rules show the relations among its concepts [CT02]. The relations among the concepts can then be used for learning conceptual structures [SSC97], which can be used to build ontology for a specific domain [MS00] or the Semantic Web [MS01]. Association rule mining for textual database can also be used for other text mining tasks such as clustering. In [HKK97, MHB+97], frequent itemsets generated are used to form a hypergraph, which is then clustered using a hypergraph partitioning algorithm. This approach is proved effective in clustering web documents.

2.3.2 Associative Classification for Text Mining

Associative classification can be applied to textual database, from which association rules can be extracted. In [AZ02, ZA02], Antonie and Zaiane proposed the *Association Rule based Categorizer* (ARC) for classification and categorization of textual documents. The algorithm is similar to CBA, which uses the Apriori-based approach for the rule discovery process, and CMAR, which uses multiple rules for classification. To classify a data sample, all rules that match it are collected and grouped according to their classes. Each class is then weighted using rules in the corresponding group, and the data sample is classified into the class that corresponds with the highest weight [ZA02]. The use of associative weighting based on association rules can also help solve the categorization problem, in which a data sample may be classified (or categorized) into one or more classes (or categories) [AE99] instead of only one class. To categorize a data sample, several categories that have the highest weights can be selected [AZ02]. The performance of this approach is comparable to other popular classification approaches, such as C4.5 and Naïve Bayes [AZ02, ZA02].

In [Its01], Itskevitch proposed the Classification using Cohesion and Multiple Association Rules (COMAR) for email classification. The COMAR algorithm is adapted from CMAR for

classifying email messages into hierarchically organized folders. Similar to ARC, associative weighting is also used to classify (or categorize) an email message into several folders. The COMAR approach has performed quite effectively when compared with the Naïve Bayes classifier [Its01].

One of the considerations in using AC approaches for textual database is performance of the rule discovery process. The process of discovering a complete set of CARs is similar to the process of association rule mining. Thus, the rule discovery process also needs to deal with the problem of mining association rules discussed before. That is, the rule discovery process may be very computationally expensive and return an uneven set of discovered rules. The complex rule discovery process may lead to long runtime and require much computing resources. The uneven set of discovered rules means that relevant but infrequent items (words) may not appear in any discovered rules. This may cause performance degradation of AC.

2.4 Summary

Association rule mining is one of the most well-studied problems in data mining. Association rule mining was originated from the market basket problem. It aims to generate rules such as "if a customer purchases items X, then he or she also purchases items Y". The mining process of association rules is based on the measures of support and confidence so that only "useful" and "certain" rules are gathered.

The main task of association rule mining is to mine the set of frequent itemsets which id then used to generate association rules. Frequent itemset mining algorithms can be conducted using the breadth-first search strategy or depth-first search strategy. For a general comparison, the depth-first search based algorithms such as Eclat and FP-growth store the database in the main memory. Therefore, they are more efficient in support counting than breadth-first search based algorithms, such as Apriori and Partition. However, no algorithm completely outperforms the others, and their performance depends highly on the specific databases that are used. The challenge of association rule mining is the complexity of the frequent itemset mining process which performs exhaustive search on the huge space of itemsets. Another challenge of association rule mining is that it returns a huge set of rules that requires computing effort to store and process in the subsequent knowledge discovery processes.

In Chapter 3, we propose a new type of constraint for association rules called category-based constraint. With this constraint, users can limit interesting association rules using a category pattern which may help eliminate many unwanted rules. In addition, we also

propose Apriori^{CB} algorithm for mining association rules with the category-based constraint. By using category patterns, it can substantially reduce the number of itemsets to be processed in each pass of the Apriori algorithm. This helps reduce the memory required and speed up the mining process. In Chapter 4, we investigate effective measures for assessing association rules. Besides support and confidence, the relative confidence of implication relationship, which is captured by certainty factor (CF) [SB90], is also evaluated. Relative confidence can be used to reflect truthfully the implication relationship of association rules by removing the noisy substance of frequency from the confidence measure. Thus, the use of relative confidence overcomes the drawback of the support and confidence framework which may lead obtaining many misleading rules. This is especially useful for mining datasets with items that are unevenly distributed such as textual datasets.

Associative classification makes use of association rule mining for the discovery of class association rules for classification. The association rule mining algorithms, such as Apriori and FP-growth, can be applied to the rule discovery process of AC. The challenge of AC is the complexity of the rule discovery process. This is especially true with large datasets from which association rules are mined with small support to form accurate associative classifier. In addition, it is also necessary to adapt conventional association rule mining framework so that the set of discovered rules is appropriate for the classification task. For example, we need to determine appropriate support and confidence values, and obtain a balance among rules of different classes in order to achieve higher accuracy for AC.

In Chapter 5, we propose the *Mstep-AC* approach for the discovery of association rules for AC. This approach is able to discover more rules for AC with the same support threshold and assess them more effectively in comparison with conventional AC approaches. Thus, the proposed approach may achieve better associative classification accuracy while maintaining the rule discovery process at a level of complexity comparable to those of conventional AC approaches. Moreover, in Chapter 6, we propose the *AIS-AC* approach based on the Artificial Immune System (AIS) for the discovery of association rules for AC. The proposed approach aims to achieve an accuracy comparable to conventional AC approaches while reducing the complexity of the mining process. By searching rules in an evolutionary manner, AIS-AC is efficient in tackling the computational complexity problem of the huge search space of rules. By adapting association rule searching strategy, the proposed AIS-based approach is able to gather a set of association rules that are suitable for effective associative classification.

Association rules are mined from transactions of items. Thus, when textual documents are converted into the standard transactional database format, association rule mining and associative classification approaches can also be applied. However, textual databases also contain some special features that make association rule mining more complicated. For example, the association rule mining for a textual database may deal with a huge collection of textual documents of long transactions. And many related words tend to co-occur in the same documents. Thus, the association rule mining process for textual database may be very complex and return a large set of association rules.

In Chapter 4, we propose the associative feature selection approach for unsupervised feature selection from textual data. The proposed approach can be used for data preprocessing of textual datasets, which are generally large and contain a lot of irrelevant features. The Apriori^{CB} algorithm can also be applied when terms occurred in textual data are categorized. In addition, the Mstep-AC and AIS-AC approaches can be used effectively to tackle the complexity problem on rule discovery for associative classification in textual databases.

Chapter 3

Category-based Constraint for Association Rule Mining

Discovery of frequent itemsets is a fundamental task of association rule mining. The challenge of frequent itemset discovery is the computational complexity of searching huge space of itemsets. In practice, users are often interested only in a subset of rules, whose items satisfy a given constraint. For example, a user of a market basket database may just want to find out rules involving items on *coffee* and *tea*. The association rule mining process then needs only to deal with the transactions containing *coffee* and *tea* which are much fewer than the whole dataset. The mining process, therefore, is simpler.

In this chapter, we propose a specific type of constraint called *category-based constraint.* The constraint stipulates the category pattern of itemsets. This kind of constraint can be used with any datasets, in which the data is categorized. In addition, we also propose an algorithm, called *category-based Apriori algorithm* (Apriori^{CB}), for mining frequent itemsets with the category-based constraint based on the Apriori [AS94] algorithm. The proposed Apriori^{CB} algorithm is fundamentally different from other association rule mining algorithms in that, instead of using sets of items, it is based on lists of items. As the number of sub-elements of a list is much smaller than that of a set, the proposed approach is much faster than other Apriori-based algorithms.

The rest of this chapter is organized as follows. In the next section, we review the related work on constraint-based association rule mining. Section 3.2 then introduces the category-based constraint. The Category-based Apriori algorithm, Apriori^{CB}, for frequent itemset mining, is discussed in Section 3.3. In Section 3.4, we discuss the use of the proposed Apriori^{CB} algorithm for association rule mining under another kind of constraint known as

the succinct constraint. Section 3.5 evaluates the performance of the proposed algorithm. Finally, Section 3.6 summarizes of the chapter.

3.1 Association Rule Mining with Constraint

Many algorithms have been proposed for mining association rules effectively [AS94, SON95, ZPOL97, HP00]. However, there are still computational problems related to the large number of frequent itemsets and the size of frequent itemsets [Heg03, AY98]. These problems are given as follows:

- One often finds that mining algorithms produce a large number of rules, most of them are uninteresting. Many interesting rules have high confidence but low support values. However, with a low support threshold (to ensure that interesting rules will not be missed), one may find that the number of found rules is unmanageable.
- The computational problem also relates to the size of frequent itemsets. In the Apriori algorithm, for every frequent itemset, we need to find the supports of all its subsets before determining whether the itemset is a candidate. Thus, for a k-itemset, the support of all 2^k subsets will have to be determined. It means that with a 20-itemset, there are more than one million subsets need to be processed.

To tackle the problems, users may define constraint in order to select only the rules they are interested in. The constraint can then be used to limit the search space, or reduce the complexity of the mining process of association rules.

Definition 3.1 (Constraint on Association Rules)**.** A constraint C on association rules is a predicate on the set of association rules $RS = \{R \mid R \text{ is an association rule}\}$, i.e., $C : RS \rightarrow \{true, false\}$. A rule R *satisfies* a constraint C if and only if $C(R) = true$. If R does not satisfy C (i.e., $C(R) = false$), we say R *violates* C.

Generally, constraint would be applied to both antecedence and consequence of association rules. Depending on how the constraint is applied, there are *single variable* or *two variable* constraint [NLHP98]. In single variable (or 1-*var*) constraint, the constraint on items can be applied separately to the antecedence and consequence of rules. For example, constraint C on rule $R : X \Rightarrow Y$ may be considered independently on antecedence and consequence under two 1-*var* constraints C_1 and C_2 such that $C(R) = C_1(X) \wedge C_2(Y)$. In two variable

(or 2-*var*) constraint, the constraint on items can be applied jointly to the antecedence and consequence of rules. For example, $C(R) = C_3(X, Y)$.

The main consideration of association rule mining with constraint is to incorporate the constraint into frequent itemset mining which is the fundamental step of the two-step procedure in association rule mining discussed in Chapter 2. We can observe that 1-var constraint can be exploited directly into frequent itemset mining. Note that if $Z = X \cup Y$ is frequent, then X and Y are also frequent. Mining association rule R with the constraint $C(R) = C_1(X) \wedge C_2(Y)$, therefore, can start with mining frequent itemsets X and Y that satisfy constraints C_1 and C_2 respectively.

The 2-var constraint cannot be exploited directly into frequent itemset mining. However, there are some possible approaches to deal with this kind of constraint. One possibility is to convert the 2-var constraint C_3 on X and Y into 1-var constraint C_4 on $Z = X \cup Y$ as follows:

$$C_4(Z) = \exists X, Y : (Z = X \cup Y) \wedge (X \cap Y = \emptyset) \wedge C_3(X, Y) \tag{3.1}$$

Lakshmanan *et al.* [LNHP99] have shown that some 2-var constraints could be efficiently reduced to 1-var constraints.

Definition 3.2 (Constraint on Itemsets)**.** A constraint C on itemsets is a predicate on the powerset 2^I of the set I of items, i.e., $C : 2^I \rightarrow \{true, false\}$. An itemset S *satisfies* a constraint C if and only if $C(S) = true$. If S does not satisfy C (i.e., $C(S) = false$), we say S *violates* C.

Let $SAT_C(I)$ be the set of itemsets in I that satisfies C. If an itemset S satisfies C, then it is equivalent to say that $S \in SAT_C(I)$. Mining frequent itemsets with the support threshold *minSupport* and item constraint C is to find all itemsets S such that $(supp(S) \geq minSupport) \wedge C(S)$, or $(supp(S) \geq minSupport) \wedge (S \in SAT_C(I))$.

Constraint was first incorporated into association rule mining in [SVA97], in which a constraint is considered as a logical expression. In [NLHP98, LNHP99, PHL01], different constraints were classified and the mining algorithms for each constraint class were given. Ng *et al.* [NLHP98] introduced two classes of constraints, namely *anti-monotone* and *succinct*. A larger class of constraints called *convertible* was also discussed in [PHL01]. In the following subsections, we review the anti-monotone constraint, succinct constraint, convertible constraint and constraint-based mining algorithms.

3.1.1 Anti-monotone Constraint

A simple approach for constraint-based rule mining is first to find all the frequent itemsets and then discard those that do not satisfy the constraint. However, the computational cost remains the same as that of the traditional mining problem. A more efficient approach is to use the constraint in each pass of the mining process to prune unsatisfied itemsets. This will considerably reduce the number of candidate itemsets for each pass. However, it may lead to a wrong result. Let's consider an example of finding rules containing items *coffee* and *tea*. Obviously, there is no 1-*itemset* containing both *coffee* and *tea*. If the constraint is applied to the first pass, then L_1 will be empty and the process terminates. This is because the constraint does not comply with the Apriori condition. A constraint C can be obtained in each pass of the mining process of an Apriori-based algorithm if it has a property similar to that of frequent itemsets. That is, with A and B as sets of items, if A does not satisfy C and B is a superset of A, then B does not satisfy C. This is equivalent to $\neg C(A) \wedge (A \subseteq B) \Rightarrow \neg C(B)$. This constraint was introduced in [NLHP98] as anti-monotone.

Definition 3.3 (Anti-monotone Constraint)**.** A constraint C is an *anti-monotone constraint* if for any itemsets A and B, $\neg C(A) \wedge (A \subseteq B) \Rightarrow \neg C(B)$.

Thus, if an itemset satisfies an anti-monotone constraint, then any subsets of it also satisfy the constraint. Obviously, frequency is an example of anti-monotone constraint. The constraint based on itemsets containing both *coffee* and *tea* is not an anti-monotone constraint. For example, itemset {*coffee, tea*} satisfies the constraint but none of its subsets satisfy the constraint.

3.1.2 Succinct Constraint

An interesting constraint called *succinct* was defined in [NLHP98] as a constraint, in which the set of all itemsets satisfied can be succinctly characterized.

Definition 3.4. A constraint C on items is *succinct* provided that $SAT_C(I)$ is a succinct powerset [NLHP98, PHL01]:

- An itemset $I_S \subseteq I$ is a *succinct set* if it can be expressed as $\sigma_p(I)$ for some selection predicates p where σ is the selection operator.
- $SP \subseteq 2^I$ is a *succinct powerset* if there exists a fixed number of succinct sets $I_1, ..., I_k \subseteq I$ such that SP can be expressed in terms of the strict power-set of $I_1, ..., I_k$ using the union and minus operators.

A succinct constraint C can be expressed under MGF (Member Generating Function) [NLHP98] in the form of $MGF_C = \{X_1 \cup ... \cup X_n \mid X_i \subseteq \sigma_{p_i}(I), 1 \leq i \leq n, \text{and } \exists k \leq n : X_j \neq \emptyset, 1 \leq j \leq k\}$ for some $n \geq 1$ and some selection predicates $p_1, ..., p_n$.

The constraint on itemsets containing *coffee* and *tea* is succinct with $MGF : \{X_1 \cup X_2 \cup X_3 \mid (X_1 \subseteq \{coffee\}) \wedge (X_1 \neq \emptyset) \wedge (X_2 \subseteq \{tea\}) \wedge (X_2 \neq \emptyset) \wedge (X_3 \subseteq I)\}$. Succinct constraint gives the component form of itemsets. For example, if we need to examine the association between items of *Drink* and *Food* which are categories of items (for example, *coffee* and *tea* are *Drink* items), the MGF will be $\{X_1 \cup X_2 \mid (X_1 \subseteq Drink) \wedge (X_1 \neq \emptyset) \wedge (X_2 \subseteq Food) \wedge (X_2 \neq \emptyset)\}$.

3.1.3 Convertible Constraint

The convertible constraint was proposed in [PHL01] which is based on the concept of prefix itemset. The definitions of prefix itemset and convertible constraint are given as follows.

Definition 3.5 (Prefix Itemset)**.** Given an order O over the set I of items, an itemset $S' = \{i_1, i_2, ..., i_l\}$ is called a *prefix itemset* of $S = \{i_1, i_2, ..., i_m\}$ with respect to O if $l \leq m$ and items in both S and S' are listed according to the order O.

Definition 3.6 (Convertible Constraint)**.** A constraint C is *convertible anti-monotone* provided that there exists an order O on items such that whenever an itemset satisfies C, so does any prefix of S. It is *convertible monotone* provided there exists an order O on items such that whenever an itemset violates C, so does any prefix of S. A constraint is *convertible* whenever it is convertible anti-monotone or convertible monotone.

The concept of prefix itemset is similar to the concept of subset except that the former is applied only to a specific order O. For example, let's consider the set of items $I = \{1, 2, 3\}$ with the order O as the increasing order of the items (i.e., $O = [1, 2, 3]$). Itemset $\{1, 2, 3\}$ then may have three subsets of two items which are $\{1, 2\}$, $\{1, 3\}$ and $\{2, 3\}$, but it has only one prefix itemset of two items which is $\{1, 2\}$.

The tight condition of prefix itemset helps the convertible constraint cover a large class of constraints. For example, let's consider the constraint $C(S) = (avg(S) \leq A)$ where $avg(S)$ is the average value of the items in S and A is a constant. Obviously, C is not an anti-monotone constraint as the average value of a subset S' of S can be greater than that of S (i.e., $avg(S') > avg(S)$). This means that S' may violate C (i.e., $avg(S) > A$) while S satisfies C (i.e., $avg(S) \leq A$). However, let's consider the order O as the increasing order of the value of items. Then, a prefix itemset of S contains the smallest items in S. This means

that the average value of a prefix itemset S' of S will always be less than or equal to that of S. Thus, if S satisfies C, then S' will also satisfy C. In other words, C is an anti-monotone convertible constraint.

3.1.4 Constraint-based Mining Algorithms

The CAP algorithm [NLHP98] was introduced to mine association rules with anti-monotone and succinct constraints. The algorithm exploits the anti-monotone constraint in the generate and count framework based on the Apriori algorithm. According to the discussion in Section 3.1.1, the anti-monotone constraint complies with the Apriori condition, and it can be used to prune unsatisfied itemsets in Apriori-based algorithms. That is, in each pass, a generated candidate itemset is pruned if it violates the anti-monotone constraint before the counting is carried out. Thus, similar to the support constraint, the anti-monotone constraint can help reduce the number of candidate itemsets and frequent itemsets in each pass. In other words, the mining process is faster with simpler itemset generation and counting operations.

Let's consider the succinct constraint. The MGF form of a constraint C can be decomposed into MGF of constraints C_α and C_β as $MGF_{C_\alpha} = X_1 \cup ... \cup X_k$ where $X_i \neq \emptyset$ with $i = 1, ..., k$ and $MGF_{C_\beta} = X_{k+1} \cup ... \cup X_n$. In the CAP algorithm, the first k passes are conducted to find frequent itemsets L_k of k items satisfying C_α. After that, C_i and L_i $(i = k+1, k+2, ...)$ are computed per normal, with a simple modification in candidate generation of the Apriori algorithm. Note that a succinct constraint is not necessary an anti-monotone constraint. Thus, the modification in the process of generating candidate itemsets in each pass of the Apriori algorithm is needed to avoid missing any frequent itemsets in subsequent passes as illustrated in the example of finding rules containing *coffee* and *tea* given in Section 3.1.1.

The convertible constraint imposes an order O on a set of items. Thus, association rule mining algorithms, which employ the depth-first search strategy, can be used to search for itemsets under the anti-monotone convertible constraint. The depth-first search process will search for itemsets following the order O so that the prefix itemsets of an itemset S will be processed earlier than S. That is, if any of the prefix itemsets of S violate the constraint, then S can be pruned.

The PIC algorithm [PHL01] was introduced to mine association rules with the convertible constraint. The algorithm applies the depth-first FP-growth [HP00] algorithm for mining with the convertible constraint. The performance result has shown that the approach can speed

Table 3.1: An example of items and categories.

Category	Drink	Grain	Dairy
Item	coffee, tea	rice	milk, yogurt

up FP-growth considerably in mining frequent itemsets with the convertible anti-monotone constraint in comparison with the conventional FP-growth algorithm without the constraint. However, the algorithm has similar performance on mining frequent itemsets with and without the convertible monotone constraint as the constraint may not be applied to prune itemsets. The reason is given as follows. According to Definition 3.6, if an itemset violates a convertible monotone constraint, then all prefix itemsets of it also violate the constraint. Thus, the prefix itemsets of an itemset can be pruned if this itemset violates the constraint. However, to minimize the search space, most of the association rule mining algorithms conduct searching from itemsets with the least number of items to itemsets with the most number of items. This means that a prefix itemset will be processed before the itemset itself. In other words, the convertible monotone constraint may not be used to prune itemsets. Nevertheless, this constraint can be used to save the computational cost of constraint checking after the mining process is completed. That is, if a prefix itemset of an itemset satisfies a monotone constraint, then this itemset needs not be checked on the constraint as it will surely satisfy it.

3.2 Category-based Constraint

Assume that we have a set of items that is categorized with some categories. Each item then belongs to one or more categories. An example is given in Table 3.1. As shown in Table 3.1, for example, items *coffee* and *tea* belong to the category *Drink*.

As an item belongs to some categories, an itemset may also "belong" to some sets of categories. For example, we may consider that itemset {*coffee, rice, milk*} belongs to the set of categories of {*Drink, Grain, Dairy*} as there is a corresponding category in the set of categories for each item in the itemset. Note that an itemset may belong to more than one set of categories as an item can belong to some categories. Conversely, a set of categories may have some itemsets that belong to it. The set of categories can then be considered as a pattern of itemsets - each itemset contains items belonging to the categories in the pattern. For example, with the category pattern of {*Drink, Grain, Dairy*}, the corresponding itemsets would be {*coffee, rice, milk*}, {*coffee, rice, yogurt*}, {*tea, rice, milk*} and {*tea, rice,*

yogurt}. These itemsets could be the items for a housewife to pick up in a supermarket when she prepares for meals with ingredients on *Drink*, *Grain* and *Dairy*. Therefore, the set of categories can be used as constraint for selecting itemsets that follow the category pattern.

Here, we will define a new kind of constraint based on the idea of category pattern called *category-based constraint*. This pattern will be a list of categories (instead of set) so that a category may occur twice or more in the pattern. The representation of a list is also convenient for the mining algorithm with this kind of constraint.

Let $I = \{i_1, ..., i_M\}$ be a set of M distinct literals called *items* and $T = \{c_1, ..., c_N\}$ be the set of N categories of items in I. We represent a category as the set of items that belong to the category: $c = \{i \mid i \subseteq I \text{ and } i \text{ belongs to } c\}$. Then, item i belongs to category c if and only if $i \in c$. For the example given in Table 3.1, we have categories and the corresponding items as $Drink = \{coffee, tea\}$, $Grain = \{rice\}$ and $Dairy = \{milk, yogurt\}$.

Definition 3.7 (Category Constraint)**.** A category-based constraint C is represented as a list of categories $C = [c_{r_1}, ..., c_{r_s}]$, with $1 \leq r_j \leq N$ for $j = 1, ..., s$.

Definition 3.8 (Itemsets Satisfying a Constraint)**.** An itemset S satisfies constraint $C = [c_{r_1}, ..., c_{r_s}]$ if there exists an order of items in $S : [i_{t_1}, ..., i_{t_s}]$ such that $i_{t_j} \in c_{r_j}$ with $j = 1, ..., s$.

For the example given in Table 3.1, the itemsets satisfying constraint [*Drink, Grain*] would be {*coffee, rice*} and {*tea, rice*}. And constraint [*Dairy, Dairy*] has only one itemset {*milk, yogurt*} that satisfies it.

The category-based constraint is usually observed in tabular databases. For example, in a table of human resources information with attributes of {*Name, Education, Occupation, Sex*}, one may be interested in relations among some attributes, for instance, *Education* and *Occupation*, or *Sex*, *Occupation* and *Education*. The category-based constraint can be expressed as [*Education, Occupation*] and [*Education, Occupation, Sex*] respectively. In another example, one may need to find out the relations among some given classes. For instance, in a mining task for textual documents of news about Middle East, one may extract terms and classify them into groups such as *Country* = {*US, England, Iran, Iraq*}, *Organization* = {*UN, IAEA, OPEC, NATO*} and *Conflict* = {*warning, protest, attack, bombing*}. News on conflicts happening in the area may be found under constraint [*Country, Country, Conflict*] or [*Country, Organization, Conflict*].

3.3 Category-based Apriori Algorithm

We define the *sub-list* relation, the *length* and *set* values, and the *plain* and *frequent* properties of a list. Note that list $[l_e, ..., l_f]$ with $e \leq f$ denotes a list of elements with continuous indexes from e to f (e.g., $[l_3, ..., l_6] = [l_3, l_4, l_5, l_6]$).

- *Sub-list* ($\subseteq$). $L_1 \subseteq L_2$ with $L_2 = [l_1, ..., l_k]$ if and only if $\exists e, f$ such that $1 \leq e \leq f \leq k$ and $L_1 = [l_e, ..., l_f]$.
- The *length* of a list is the number of elements of that list. For example, let $L = [l_1, ..., l_k]$, the *length* of L is k. L is called a *k-list*.
- *Plain list* is a list with no repetition of elements. If $L = [l_1, ..., l_k]$ is a plain list, then $\forall e, f : (1 \leq e, f \leq k) \wedge (e \neq f) \Rightarrow l_e \neq l_f$.
- The *set of a list* is the set of all elements of that list. If $L = [l_1, ..., l_k]$, then the set of L is $\{l_1\} \cup ... \cup \{l_k\}$. If L is a *plain* list, then the set of L is $\{l_1, ..., l_k\}$ (the set of a plain *k-list* is a *k-set*).
- Let L be a list of items. L is *frequent* if and only if the set of it is a frequent itemset.

We define the *belongs-to* and *sub-belongs-to* relations on lists of items and lists of categories as follows. Note that A and B are lists of items, and M and N are lists of categories.

Definition 3.9 (belongs-to)**.** *A belongs-to M* if and only if $length(A) = length(M) = k$ and the j^{th} item of A belongs to j^{th} category of M with $j = 1, ..., k$ (i.e., $A = [i_1, ..., i_k]$; $M = [c_1, ..., c_k]$ and $i_j \in c_j$ for all $j = 1, ..., k$).

Definition 3.10 (sub-belongs-to)**.** *A sub-belongs-to N* if and only if there exists M such that *A belongs-to M* and $M \subseteq N$.

For example, [*tea, rice*] *belongs-to* [*Drink, Grain*]. Besides, [*tea, rice*] *sub-belongs-to* [*Drink, Grain, Dairy*] since [*tea, rice*] *belongs-to* [*Drink, Grain*] and [*Drink, Grain*] is a sub-list of [*Drink, Grain, Dairy*]. Our definitions above also imply that if *A sub-belongs-to N*, and A and N have the same *length*, then *A belongs-to N*.

Lemma 3.1 (sub-belongs-to is Anti-monotone)**.** Let C be a list of categories. *Sub-belongs-to C* is an anti-monotone constraint on item-lists with the relation *sub-list*: (*B sub-belongs-to C*) $\wedge$ ($A \subseteq B$) $\Rightarrow$ (*A sub-belongs-to C*).

Proof. Let $B = [i_1, ..., i_k]$. $A \subseteq B \Rightarrow \exists e, f$ such that $1 \leq e \leq f \leq k$ and $A = [i_e, ..., i_f]$. B *sub-belongs-to* $C \Rightarrow \exists N$ such that B *belongs-to* N and $N \subseteq C$. From the definition of *belongs-to* we have $N = [c_1, ..., c_k]$ and $i_j \in c_j$ for all $j = 1, ..., k$. Let $M = [c_e, ..., c_f]$, we can imply that A *belongs-to* M and $M \subseteq N$. $M \subseteq N$ and $N \subseteq C$ imply $M \subseteq C$. Combining A *belongs-to* M with $M \subseteq C$, we have A *sub-belongs-to* C. □

It can be observed from Definition 3.7 and Definition 3.8 that an itemset S satisfies constraint $C = [c_{r_1}, ..., c_{r_s}]$ if and only if there exists a *plain* list L such that S is the set of L and L *belongs-to* C. L *belongs-to* C can be expressed as L *sub-belongs-to* C and $length(L) = |C|$. Now, the problem of mining frequent itemsets with category-based constraint C can be stated as finding all lists with each list L satisfying the following conditions:

- L is *frequent* and *plain*. (3.2)
- L *sub-belongs-to* C. (3.3)
- $length(L) = |C|$. (3.4)

According to Lemma 3.1, condition (3.3) is an anti-monotone constraint. Therefore, as discussed in Section 3.1.1, this constraint can be exploited in the Apriori algorithm. The Category-based Apriori algorithm (Apriori^{CB}) is given in Algorithm 3.1. The main difference between Apriori and Apriori^{CB} lies in the candidate generation process, which is carried out by the *Gen* function. In Apriori^{CB}, the *Gen* function is modified so that only lists that *sub-belongs-to* C are generated. The main process is also modified for processing frequent and candidate lists (instead of sets) of items in each pass. Condition (3.4) can simply be obtained by getting only the result of pass s (L_s with $s = |C|$) instead of combining the results of all passes $\bigcup_k L_k$. The *plain* property of L is guaranteed by the *Gen* function.

In the *Gen* function of the Apriori algorithm, a k-itemset is generated as a candidate if all of the k subsets of size (k - 1) are frequent. When lists of items are used instead of itemsets, as a k-list $L = [e_1, ..., e_k]$ has only two sub-lists of $(k-1)$ items (i.e., $L_1 = [e_1, ..., e_{k-1}]$ and $L_2 = [e_2, ..., e_k]$ which are called *immediate* sub-lists of L), a k-list is generated as a candidate if its two immediate sub-lists are frequent.

Let $R_{ki} = [c_{r_i}, ..., c_{r_{(k+i-1)}}]$ be the sub-list of length k and start at the i^{th} category of constraint $C = [c_{r_1}, ..., c_{r_s}]$. The two immediate sub-lists of R_{ki} should have the length of $(k-1)$, and start at the i^{th} and $(i+1)^{th}$ categories of C. Therefore, they are $R_{(k-1)i}$ and $R_{(k-1)(i+1)}$. According to Definition 3.8, any candidate k-list *belongs-to* a sub-list of length k

Algorithm 3.1 Category-based Apriori.

Input:

D - training dataset
$C = [c_{r1}, ..., c_{rs}]$ - category-based constraint
$minSupport$ - minimum support threshold

Output:

L - set of frequent itemsets

Process:

1: **for** $i = 1$ **to** s **do**
2: $C_{1i} = C_{ri}$
3: **end for**
4: $L_{1i} = \{l \mid (l \in C_{1i}) \wedge (supp(l) \leq minSupport)\}$
5: $k = 2$;
6: // pass k
7: **while** $(L_{(k-1)i} \neq \emptyset \ \forall i = 1, ..., s - (k-1) + 1) \wedge (k \leq s)$ **do**
8: Gen(k)
9: **for** $i = 1$ **to** $s - k + 1$ **do**
10: $L_{ki} = \{l \mid (l \in C_{ki}) \wedge (supp(s) \geq minSupport)\}$
11: **end for**
12: $k = k + 1$
13: **end while**
14: **return** $L = L_{s1}$
15:
16: function Gen(k)
17: **for** i $= 1$ **to** $s - k + 1$ **do**
18: **insert into** C_{ki}
19: **select** $p.item_1, p.item_2, ..., p.item_{k-1}, q.item_{k-1}$
20: **from** $L_{(k-1)i}\ p, L_{(k-1)(i+1)}\ q$
21: **where** $(p.item_2 = q.item_1) \wedge ... \wedge (p.item_{k-1} = q.item_{k-2}) \wedge (p.item_1 \neq q.item_{k-1})$
22: **end for**

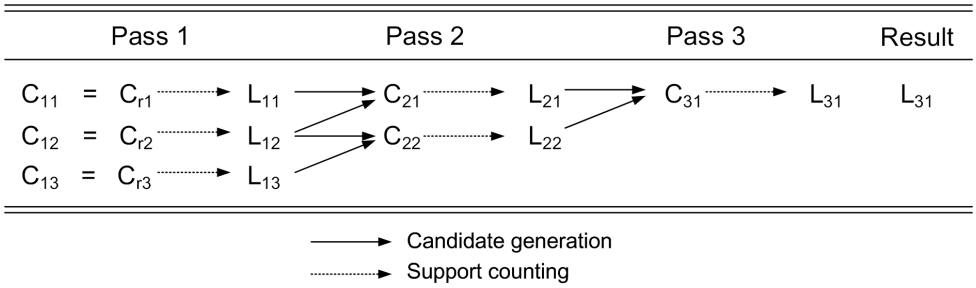


Figure 3.1: Candidate generation and counting with $C = [C_{r_1}, C_{r_2}, C_{r_3}]$.

of C because it *sub-belongs-to* C. Let $A = [e_1, ..., e_k]$ be a candidate list of items in pass k such that A *belongs-to* R_{ki}. A is generated in the *Gen* function from its two immediate sub-lists: $A_1 = [e_1, ..., e_{(k-1)}]$ and $A_2 = [e_2, ..., e_k]$. We can imply from the definition of sub-list that A_1 and A_2 *belongs-to* the two sub-lists of R_{ki} accordingly (with A_1 *belongs-to* $R_{(k-1)i}$ and A_2 *belongs-to* $R_{(k-1)(i+1)}$).

With $k \leq s$, constraint C has $t = s - k + 1$ sub-lists of size k: $R_{k1} = [c_{r_1}, ..., c_{r_k}], ..., R_{k_t} = [c_{r_t}, ..., c_{r_s}]$. In pass k, candidate lists of items are stored in t distinct variables $C_{k1}, ..., C_{kt}$ such that candidate lists of items in C_{ki} *belongs-to* sub-list R_{ki} of C with $i = 1, ..., t$. Frequent k-item-lists are also stored accordingly in $L_{k1}, ..., L_{kt}$. We can imply that candidate list A of items is stored in C_{ki}, and its sub-lists A_1 and A_2 are stored in $L_{(k-1)i}$ and $L_{(k-1)(i+1)}$ respectively. It means that candidate k-item-lists in C_{ki} are generated from frequent $(k-1)$-item-lists in $L_{(k-1)i}$ and $L_{(k-1)(i+1)}$. An example of candidate generation and frequent itemset counting is given in Figure 3.1 with $C = [C_{r_1}, C_{r_2}, C_{r_3}]$.

Let's consider a scenario of mining frequent itemsets with a category-based constraint using a conventional Apriori-based algorithm. To obtain a frequent itemset satisfying the constraint, an Apriori-based algorithm generates and counts the support of every subset of the itemset. For example, with the constraint of size 10 (i.e., the size of the itemset is also 10), there would be at least $2^{10} - 1 = 1023$ subsets of the itemset (except the empty one) to be processed. In Apriori^{CB}, however, there are only 55* sublists of list size 10. This implies that the number of itemsets processed in Apriori^{CB} would probably be much smaller than that of an Apriori-based algorithm. In other words, Apriori^{CB} would be faster in comparison with a Apriori-based algorithm and the difference in runtime between them would get larger when the size of the constraint increases. The reason is that the proportion of the number of

*There are C_{10}^1 sub-lists of one item and C_{10}^2 sub-lists with more than one item, each of which corresponds to a pair of distinct first and last items. Therefore, the number of sub-lists of a list of 10 items is $C_{10}^1 + C_{10}^2$.

sub-elements of lists to that of sets decreases exponentially when the size of itemsets increases. For example with a 20-itemset, the proportion would be $\frac{C_{20}^1+C_{20}^2}{2^{20}-1} \approx \frac{1}{5000}$.

Next, let's consider memory usage of different approaches including Apriori and FP-growth. Generally, the memory usage in Apriori depends on the size of the hash tree in each pass while in FP-growth, it is determined by the size of the FP-tree. Thus, Apriori may need to use a large amount of main memory to construct the hash tree if there are a large number of candidate itemsets in a certain pass during the mining process. On the other hand, FP-growth may require a large amount of memory, which is used to construct the FP-tree, to mine association rules from a large dataset. The small number of candidate itemsets in Apriori$^{\text{CB}}$ leads to a small-sized hash tree to be constructed in each pass. Thus, the amount of main memory used in Apriori$^{\text{CB}}$ would be relatively quite small. This means that Apriori$^{\text{CB}}$ may not suffer from the problem of large memory usage caused by a large amount of candidate itemsets as in Apriori or a large FP-tree as in FP-growth.

Above discussion suggests that the use of Apriori for the implementation of Apriori$^{\text{CB}}$ helps reduce the memory usage substantially. This is, however, not the case if the FP-growth is applied as the same FP-tree is still constructed. In addition, the Apriori-based algorithm allows full exploitation of the anti-monotone constraint in the mining process (e.g., the *sub-belongs-to* constraint). This helps optimally limit the number of candidates, thereby minimizing the number of itemsets to be processed.

3.4 Category-based vs. Succinct

Category-based and succinct constraints are similar in terms of specifying the "ingredient pattern" of itemsets. However, the scope that the category-based constraint focuses on is narrower than that of the succinct constraint. For example, with category-based constraint $A = [Drink, Grain]$, one may have $\{coffee, rice\}$ or $\{tea, rice\}$. The succinct constraint B that focuses on both *Drink* and *Grain* items may get $\{coffee, rice\}$, $\{tea, rice\}$ or $\{coffee, tea, rice\}$. The restriction on "one item for each category" helps narrow down the scope of the itemsets that satisfy A.

A natural question to ask is that why we need to define a new category-based constraint A when one can use a succinct constraint B to mine all itemsets satisfying A. The answer could be very simple that we may be interested only in itemsets satisfying A so that if we use B we will waste time to mine itemsets we are not interested in. In addition, the category-based constraint can be used for mining with the succinct constraint. Figure 3.2 illustrates how

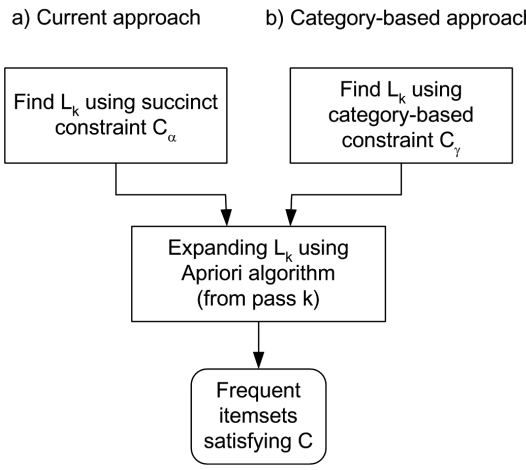


Figure 3.2: Mining association rules with succinct constraint C.

to use the category-based constraint to speed up the process of mining under the succinct constraint C. In Figure 3.2, constraints C_α, C_β, C_γ are generated from C. This is discussed as follows.

Let's look at the MGF form of a succinct constraint C and the MGF forms of C_α and C_β which are decomposed from MGF_C:

$$MGF_C = \{X_1 \cup ... \cup X_n \mid X_i \subseteq \sigma_{p_i}(I), 1 \leq i \leq n, \text{ and } \exists k \leq n : X_j \neq \emptyset, 1 \leq j \leq k\} \quad (3.5)$$

$$MGF_{C_\alpha} = \{X_1 \cup ... \cup X_k \mid X_i \subseteq \sigma_{p_i}(I), \text{ and } X_j \neq \emptyset, 1 \leq j \leq k\} \quad (3.6)$$

$$MGF_{C_\beta} = \{X_{k+1} \cup ... \cup X_n \mid X_i \subseteq \sigma_{p_i}(I)\} \quad (3.7)$$

An itemset S satisfying constraint C can then be decomposed into two sets S_1 and S_2 satisfying constraints C_α and C_β respectively (i.e., $S = S_1 \cup S_2$). Note that condition $X_i \neq \emptyset$ means that S_1 contains at least one item in each $\sigma_{p_i}(I)$ with $i = 1, ..., k$. Thus, S_1 can be decomposed further into two sets S_1' and S_1'' such that $S_1 = S_1' \cup S_1''$, where S_1' contains exactly one item in each $\sigma_{p_i}(I)$ with $i = 1, ..., k$. Similarly, S_1' and S_1'' satisfy respectively constraints $C_{\alpha'}$ and $C_{\alpha''}$ decomposed from C_α as follows:

$$MGF_{C_{\alpha'}} = \{X_1 \cup ... \cup X_k \mid X_i \subseteq \sigma_{p_i}(I), \text{ and } |X_i| = 1, 1 \leq i \leq k\} \quad (3.8)$$

$$MGF_{C_{\alpha''}} = \{X_1 \cup ... \cup X_k \mid X_i \subseteq \sigma_{p_i}(I)\} \quad (3.9)$$

Let's consider an itemset T of k items satisfying constraint C_α. According to the MGF of C_α, T contains at least one item in each $\sigma_{p_i}(I)$ with $i = 1, ..., k$. This can imply that T

contains exactly one item in each $\sigma_{p_i}(I)$ with $i = 1, ..., k$ (otherwise T must contain more than k items). This means that T also satisfies constraint $C_{\alpha'}$. In the CAP algorithm, the first k passes are conducted to find frequent itemsets L_k of k items satisfying C_α. Therefore, the algorithm actually searches for frequent itemsets L_k of k items satisfying $C_{\alpha'}$ in the first step.

Let C_γ be the category-based constraint such that $C = [\sigma_{p_1}, \sigma_{p_2}, ..., \sigma_{p_k}]$. Itemsets that found under this constraint will contain one element for each $\sigma_{p_i}(I)$ with $(i = 1, ..., k)$. In other words, category-based constraint C_γ is similar to succinct constraint $C_{\alpha'}$. Therefore, the algorithm of mining rules with the category-based constraint can be used in the first k passes of association rule mining with the succinct constraint. The performance evaluation discussed in the next section will show that the approach used for category-based constraint mining is much faster than the current approach for mining with the succinct constraint.

3.5 Performance Evaluation

We have conducted three experiments to evaluate the effectiveness of the Category-based Apriori algorithm (Apriori^{CB}) using a 3 GHz Pentium PC with 1 GB of memory running with Microsoft Windows XP. We have chosen synthetic datasets, which were generated by simulating supermarket transactions [AS94], in these experiments. The reason for choosing synthetic datasets is that it is flexible for experimenting characteristics of the datasets which help evaluate different aspects of the implemented algorithms. For example, it is possible to set the number of categories and selectivity rates to some given values before datasets are generated. We have used synthetic datasets generating tool called ARtool [Cri], which is a Java application provided with open source code, to generate synthetic datasets. In the first two experiments, two datasets have been generated using ARtool, namely Syn20 and Syn30, with parameters set similar to the those of the datasets used in [AS94] and [HPY00]. In our case, the average transaction sizes of the datasets are set to 20 and 30, respectively. The average maximal potentially frequent itemset size and the number of transactions of both datasets are set to 10 and 100,000 respectively. The other parameters of these datasets are set the same as those used in [AS94], including the number of items (i.e., 1000) and the number of maximal potentially frequent itemsets (i.e., 2000). In these synthetic datasets, categories are formed by dividing the set of items equally into groups, with each considered as a category. In these experiments, ten categories labelled as $c_1, ..., c_{10}$ have been used, each containing 100 items.

With category-based constraint $C = [c_{r1}, ..., c_{rs}]$, the constrained association rule mining algorithm CAP discussed in [NLHP98] can be applied to mine frequent itemsets. In this case, C is loosened optimally into a "weaker" constraint C_γ stipulating that an itemset $S = \{e_1, ..., e_t\}$ satisfies the constraint if (i) $t \leq s$; (ii) all items of S belong to $I_C = c_{r1} \cup ... \cup c_{rs}$; and (iii) there is a permutation $[c_{u_1}, ..., c_{u_s}]$ of C such that $e_i \in c_{u_i}$ with $i = 1, ..., t$. Here, permutation refers to a certain order of items in a list. For example, if C = [*Drink, Drink, Dairy*], then there are three permutations of C : [*Drink, Drink, Dairy*], [*Drink, Dairy, Drink*] and [*Dairy, Drink, Drink*].

It is clear that C_γ is an anti-monotone constraint, and when condition (i) is replaced by $s = t$, then the set of itemsets mined with C_γ will be the same as that with C. We can see that, similar to the anti-monotone constraint, conditions (ii) and (iii) can be used to discard itemsets which can surely not be the subsets of itemsets satisfying C. For the example given in Table 3.1, if the constraint C = [*Drink, Drink, Dairy*], then $I_C = Drink \cup Dairy$ = {*coffee, tea, milk, yogurt*}. The candidate 1-itemsets satisfying condition (ii) would be itemsets that contain items within I_C which could be {*coffee*}, {*tea*}, {*milk*} or {*yogurt*}. This means that itemsets {*rice*} are discarded. Condition (iii) discards the 2-itemset {*milk, yogurt*}, when the itemset is generated. It is because condition (iii) requires the two items to follow one of the patterns formed from the first two categories of a permutation of C which could be [*Drink, Drink*], [*Drink, Dairy*] or [*Dairy, Drink*]. As constraint C_γ is anti-monotone, it can be used in association rule mining algorithms like the support constraint. The only difference is that while the support constraint is based on support counting, the conditions in constraint C_γ (i.e., conditions (ii) and (iii)) apply the category pattern to prune itemsets.

Apart from the proposed Apriori$^{\mathrm{CB}}$ algorithm, we have also implemented the following algorithms for comparison:

- The CAP algorithm implemented with constraint C_γ based on the Apriori algorithm [NLHP98]. We call this implementation as CAP-A$^{\mathrm{CB}}$.
- The CAP algorithm implemented with constraint C_γ based on the FP-growth algorithm. We call this implementation as CAP-F$^{\mathrm{CB}}$.
- The Apriori algorithm and the FP-growth algorithm. These two algorithms are used in the third experiment.

All the above four algorithms and Apriori$^{\mathrm{CB}}$ are implemented based on Goethals' well-known implementations for frequent itemset mining [Goe02] (release date: 1/6/2003) which

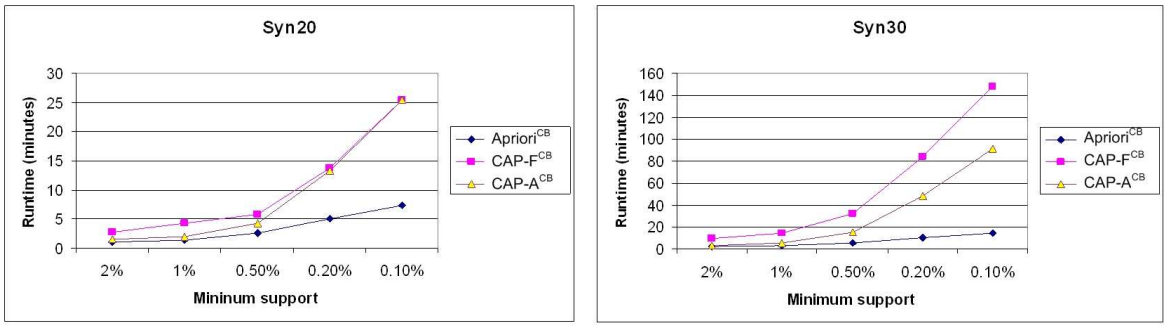


Figure 3.3: Runtime performance for CAP^{CB} and Apriori^{CB}.

are available online [Goe]. These algorithms are implemented with the aim that tries to minimize changes from the original Goethals' implementations. For example, implementations of CAP-A^{CB} and CAP-F^{CB} are done by modifying only the pruning procedure from Goethals' Apriori and FP-growth algorithms. The implementation of Apriori^{CB} is obtained by modifying the candidate generation function.

In the first experiment, we have used the category-based constraint formed from the list of all ten categories (i.e., $[c_1, ..., c_{10}]$). In this experiment, we run the three algorithms with the support threshold value varying from 0.1% to 2% (i.e., with the corresponding number of data samples from 100 to 2000). The runtime performance of the three algorithms is given in Figure 3.3. The performance result has shown that the runtime of Apriori^{CB} is much shorter than that of CAP-A^{CB} and CAP-F^{CB}, and the difference between them increases drastically when the support threshold gets smaller.

Table 3.2 gives the memory usage based on the numbers of nodes of the trees constructed in different algorithms with the support threshold value of 0.2%. In CAP-F^{CB}, the memory usage is indicated by the number of nodes of the FP-tree while in Apriori^{CB} and CAP-A^{CB}, the memory usage is shown by the number of maximal nodes of the hash trees generated during the mining process. The table has shown that the memory usage in Apriori^{CB} is substantially smaller than that in CAP-A^{CB} and CAP-F^{CB}. The large difference in memory usage of Apriori^{CB} and CAP-A^{CB} is mainly due to the small proportion on the number of sub-elements of lists to that of sets. The small proportion on the number of candidate itemsets generated in Apriori^{CB} compared with CAP-A^{CB}, in turn, leads to a small proportion between the sizes of the hash trees constructed from these two algorithms. Note that, according to the implementation, the size of a node in a hash tree, which consists of two integers to store the item label and support count, and a pointer to children nodes, is considerably smaller than

Table 3.2: Memory usage with the support threshold of 0.2%.

	Apriori^{CB}	CAP-A^{CB}	CAP-F^{CB}
Syn20	79,058	415,213	1,515,291
Syn30	316,083	2,873,988	2,580,051

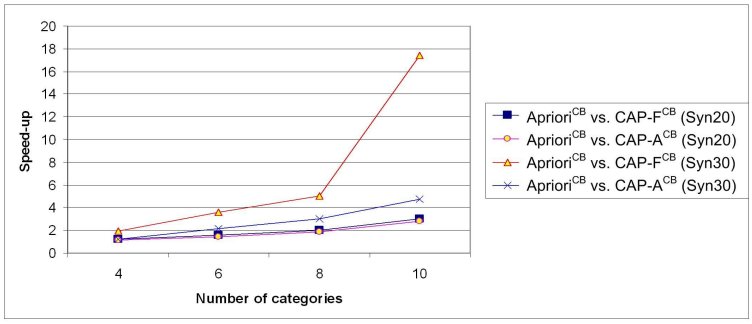


Figure 3.4: Runtime performance of Apriori^{CB} relative to other algorithms with various lengths of categories in constraint.

that of a node in FP-tree, which contains two integers to store the item label and support count, and three pointers to the node's branch, its parents and children.

In the second experiment, we study performance of different algorithms when the number of categories varies. In particular, the experiment is conducted with the category-based constraint of 4, 6, 8 and 10 categories respectively. For a category-based constraint of k categories, the first k categories in the dataset is chosen to form the category-based constraint, i.e., $[c_1, ..., c_k]$. The speed-up of Apriori^{CB} relative to CAP-A^{CB} and CAP-F^{CB} with various lengths of the category-based constraint for the support value of 0.2% is given in Figure 3.4. The performance result has shown that the runtime of Apriori^{CB} is slightly smaller than that of CAP-A^{CB} and CAP-F^{CB} when the constraint has four categories. However, Apriori^{CB} runs much faster than CAP-A^{CB} and CAP-F^{CB} when the number of categories gets larger. This was explained earlier by the small proportion of the number of sub-elements of lists to that of sets which grows exponentially with increasing size of itemsets. Thus, when the size of the constraint is increased, the difference between the number of itemsets processed in Apriori^{CB}, and CAP-A^{CB} and CAP-F^{CB} gets larger and leads to a larger difference in runtime between Apriori^{CB} and the other two algorithms.

In the third experiment, we examine performance of different algorithms on the selectivity of the constraint. Let k be the size of the category-based constraint. The selectivity of the

category-based constraint is considered as the ratio a/b, where a is the number of the frequent k-itemsets satisfying the constraint and b is the number of all available frequent k-itemsets. This experiment has two differences from the first two experiments:

- For most of the datasets, including synthetic ones, the selectivity value would be very small. For example, in the first two experiments, the number of k-itemsets satisfying the constraint is only around 1/100 to 1/1000 of the number of all frequent k-itemsets. However, with a synthetic dataset, we may control this rate by modifying the procedure of data generation.

- All three algorithms used in the first two experiments are for "constraint mining". Thus, only k-itemsets satisfying the constraint are mined with these algorithms. To study the relationship between the speed-up of Apriori^{CB} against the selectivity of the constraint, we need to compare Apriori^{CB} with the original rule mining algorithms, such as Apriori and FP-growth, which can generate the complete set of all available frequent k-itemsets.

The implementation of synthetic dataset generation in [Cri] generates data in two steps. In the first step, a random pool of potentially frequent itemsets is created. In the second step, the synthetic dataset is generated using this pool of itemsets. To generate a dataset with selectivity rate s, we have simply modified the procedure of generating the itemset pool, so that the ratio between the number of itemsets satisfying the constraint with the total number of itemsets in the pool is equal to a given selectivity rate. The selectivity on the frequent itemsets mined will be very close to this value. In this experiment, we set the selectivity rate between 20% and 100%. We name two new datasets as Syn20-sel and Syn30-sel with respect to that of the datasets, Syn20 and Syn30, used in the first two experiments.

For association rule mining algorithms, the corresponding Apriori and FP-growth implementations in [Goe] have been used. We have only made a modification of these two implementations on setting the maximal length of frequent itemsets to k, so that the algorithms will not waste time to search for frequent itemsets with length above k. The speed-up of Apriori^{CB} against Apriori and FP-growth with various values of selectivity for the support value of 0.2% and the category-based constraint of all ten categories (i.e., $[c_1, ..., c_{10}]$) is given in Figure 3.5. The performance result has shown that Apriori^{CB} can speed-up against Apriori and FP-growth considerably even with selectivity of 100% (i.e., Apriori^{CB}, and Apriori and FP-growth return the same set of frequent itemsets). This is similar for the case that

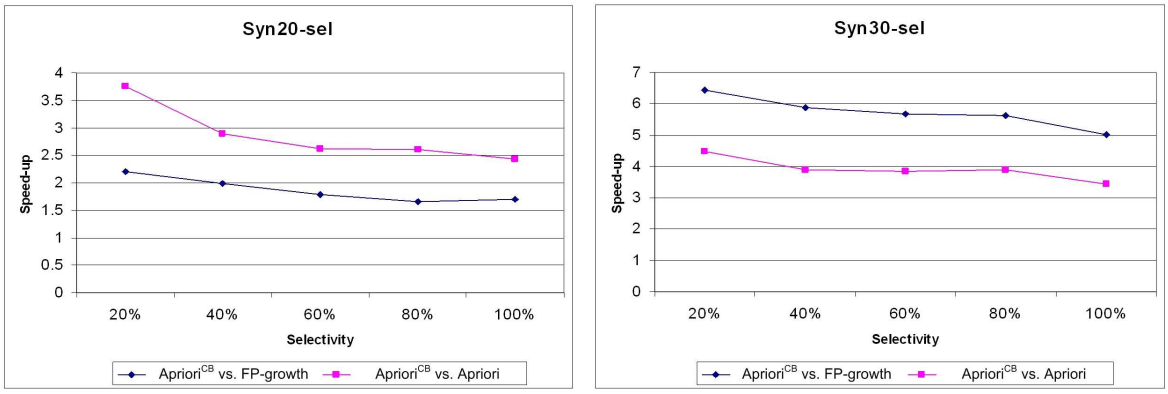


Figure 3.5: Runtime performance of Apriori^{CB} relative to other algorithms with different selectivity values.

Apriori^{CB} can speed-up against CAP-A^{CB} and CAP-F^{CB}. We observe that, according to the performance results given in Figure 3.5, Apriori^{CB} speeds up against Apriori more than that of FP-growth for the Syn20-sel dataset, but the result is reversed for the Syn30-sel dataset. It means that Apriori is slower than FP-growth for the Syn20-sel dataset, whereas it is faster for the Syn30-sel dataset.

3.6 Summary

The proposed Category-based Apriori algorithm is fundamentally different from conventional association rule mining algorithms in that, instead of using sets of items, it is based on lists of items. The precious property of lists is that the number of sub-elements is much fewer than that of sets. For example, a set of 20 items has more than one million subsets while a list of the same size has only 210 *sub-lists*. In addition, let's consider the set {1, 2, 3, 4}. The subsets of two items are {1, 2}, {1, 3}, {1, 4}, {2, 3}, {2, 4} and {3, 4}, while the list [1, 2, 3, 4] has only three sub-lists of two items: [1, 2], [2, 3] and [3, 4].

In a category-based constraint, itemsets have to contain exactly one item in each category of the category pattern. This condition can be satisfied if the user knows exactly the "pattern" of itemsets he or she is looking for. Generally, one cannot find itemsets that consist of items that he or she does not know the corresponding categories. Moreover, every category of a constraint is mandatory. In other words, it has to be present in every resultant itemset. For example, with constraint [*Drink, Grain, Dairy*], one would not get itemset {*coffee, milk*} because category *Grain* does not present in the itemset.

To make the category-based constraint more flexible, the proposed approach can be extended to find relations of some given categories with an "unknown" one by adding a category called *All* which contains every item. For example, one may want to find relations between categories *Drink* and *Grain* with an unspecified category. The constraint can then be expressed as [*Drink, Grain, All*]. In addition, the category-based constraint can also be extended by adding a *'null'* item to some categories with the convention that item *null* is contained in every transaction. In this case, with constraint [*Drink, Grain, Dairy*], by adding *null* item to category *Grain,* one may get the result of itemset {*coffee, null, milk*}. Thus, {*coffee, milk*} is an itemset under pattern [*Drink, Grain, Dairy*]. It means that category *Grain* may not need to occur in the resultant itemsets.

Chapter 4

Relative Confidence for Association Rule Mining

Association rule mining searches for interesting relationships amongst items in a given dataset. An association rule $X \Rightarrow Y$ means that whenever a transaction contains X, it will probably contain Y. The probability or confidence of rule $X \Rightarrow Y$ presents the imply relationship (or implication) from X to Y. However, one of the main drawbacks of the confidence measure is that it is influenced by the frequency of itemsets. Thus, the resultant set is biased towards association rules of very frequent itemsets. For example, if two itemsets X and Y have a very high frequency, then they will probably form a rule with a high confidence even if there is no relation between them at all.

In this chapter, we investigate effective measures for association rule mining. Besides support and confidence, certainty factor (CF) [SB90] is also considered. The relative confidence of implication relationship, which is captured by the certainty factor measure, is able to reflect truthfully the implication relations of association rules. We analyze the difference between different measures especially the confidence and relative confidence based on contexts of the market basket problem, textual data and associative classification. From our experimental results, the relative confidence (captured by the CF measure) can be used to overcome drawbacks on generating many misleading rules in the conventional support and confidence framework.

In addition, we also propose an approach called associative feature selection for text mining in this chapter. The approach is based on associative relations among items, which are represented by measures such as confidence or CF, to compute the relevant features. The proposed associative feature selection approach can also be used as a means for evaluating

Table 4.1: An example of three rules obtained from D514 (N = 514).

Rule R	R_1	R_2	R_3
Antecedence X	university	university	Mine
Consequence Y	study	data	data
Document frequency of X (a)	11	11	330
Document frequency of Y (b)	85	416	416
Document frequency of X∧Y (c)	5	5	316
P[Y] ($\frac{b}{N}$)	17%	81%	81%
supp(R) ($\frac{c}{N}$)	1%	1%	61%
conf(R) ($\frac{c}{a}$)	45%	45%	96%

the effectiveness of the proposed framework in depicting associative relations among textual data items.

The rest of this chapter is organized as follows. We first review different measures for assessing association rules in Section 4.1. Relative implication relationships of association rules and relative confidence are then discussed in Section 4.2. Section 4.3 presents the associative feature selection approach for text mining and its performance evaluation. Finally, a summary is given in Section 4.4.

4.1 Measures for Association Rules

To illustrate characteristics of different measures for association rules, let's consider the following experiment that was conducted in this research. A collection of textual documents called D514, which was extracted from published papers on data mining, was used for the experiment. The papers were obtained from searching the Science Direct Web page (http://www.sciencedirect.com/) on June 2003 with the keyword on "data mining" in their "title", "abstract" and "keyword" fields. A total of 514 papers were found and the abstracts of the papers were stored for the document collection. The size of each document varied from 2 to 27 lines of text. This document collection will also be used later for evaluating different measures for capturing implication relationship in Section 4.2.

Here, a document is considered as a transaction, and a *term* (word or phrase) is treated as an item. The *document frequency* of a term is the number of documents that contain the term. Table 4.1 shows three rules, namely R_1, R_2 and R_3, obtained from the document collection with term frequencies, and corresponding support and confidence values.

Table 4.2: Some other measures for association rules.

Measure	Formula	
Interest	$intr(R) = \frac{P[X \wedge Y]}{P[X] \times P[Y]} = \frac{conf(R)}{P[Y]}$	(4.1)
Conviction	$conv(R) = \frac{P[X] \times P[\neg Y]}{P[X \wedge \neg Y]}$	(4.2)
Reliability	$relb(R) = \frac{P[X \wedge Y]}{P[X]} - P[Y] = conf(R) - P[Y]$	(4.3)
Certainty factor	$CF(R) = \frac{conf(X) - P[Y]}{1 - P[Y]}$	(4.4)

We can observe that two rules R_1 ($university \Rightarrow study$) and R_2 ($university \Rightarrow data$) have the same values of support (1% or 5 in 514 documents) and confidence (45% or 5 in 11 documents). Obviously term *study* is closely related to *university*. *University* is for "studying" and conversely "studying", as one may wish, is preferably to be done in a *university*. The relation is also represented by the confidence measure. $conf(R_1) = 45\%$ means that 45% of the documents mentioning about *university* also contain *study*. This value is much higher than the probability of a document randomly containing *study* which is $supp(study) = 17\%$. Rule R_2 also has the same value of confidence. However, in this case, rule $university \Rightarrow data$ cannot be considered as interesting. The reason is that consequence *data* of rule R_2 is contained in 416 out of the 514 documents, or $supp(\text{data}) = 81\%$. It means that with a random term Z then $supp(Z \Rightarrow data)$ is around 81%. Thus, *university* and *data* actually have "negative implication" as $supp(university \Rightarrow data)$ is 45%, which is much less than 81%.

In addition to the conventional measures of support and confidence for association rules, other measures have also been proposed with an intention to fully characterize association rules. Table 4.2 lists some other measures including interest [BMUT97], conviction [BRS97], reliability [AEMT00] certain factor [SB90, BBSV02].

4.1.1 Interest Measure

Interest was proposed in [BMUT97] as a measure of correlation between items. The interest value of a rule R is the proportion of the confidence value of the rule with respect to the value of the consequence support of R. An interest value above 1 represents a positive correlation between two items (or itemsets) while the value below 1 represents a negative correlation. The interest value of 1 tells that the terms actually have no correlation. In the example given

in Table 4.1, $intr(university, study) = \frac{45\%}{17\%} = 2.6$ and $intr(university,\ data) = \frac{45\%}{81\%} = 0.55$. These values illustrate that there exists a positive correlation of *university* and *study,* and a negative correlation of *university* and *data*. However, interest is a poor measure for examining implications. For example, the interest measure of rule R_3 ($mine \Rightarrow data$) is relatively small (i.e., $\frac{96\%}{81\%} = 1.19$) despite the fact that *mine* is extremely related to *data* (the confidence value of rule R_3 is 96%). Even with any terms that have almost 100% implication to *data*, the value of interest will never exceed 1.23 (i.e., $\frac{100\%}{81\%} = 1.23$).

4.1.2 Conviction Measure

The *conviction* of a rule was introduced in [BRS97] as an asymmetric version of the interest measure. The formula of conviction can be rewritten as $\frac{1}{intr(X \Rightarrow \neg Y)}$. The conviction of rule R, therefore, logically reflects the dis-correlation of X and $\neg Y$. It means that the conviction of R reflects the correlation of X and $\neg(\neg Y) = Y$. In other words, the conviction measure also shows the correlation among items as the interest measure.

4.1.3 Reliability Measure

In [AEMT00], *reliability* of a rule R is defined as the difference between the value of the confidence measure and that of the consequence support of the rule. It measures the effect of the available information about the antecedence on the probability of the consequence [AEMT00]. Higher reliability implies stronger association "if X then Y". Positive correlation is indicated by a positive value of reliability while negative correlation is indicated otherwise. However, this pure difference in values does not truthfully reflect the implication as well as the correlation among the items. If supp(Y) is high, then the reliability value is low regardless of the values of confidence and interest (i.e., $relb(R) = conf(R) - P[Y] \leq 1 - P[Y]$).

4.1.4 Certainty Factor Measure

Certainty factor (CF) [SB90] was introduced in MICYN - one of the best models in the development of rule-based expert systems [BBSV02]. The certainty factor $CF(h|e)$ measures the belief in a hypothesis h based on an evidence e:

$$CF(h|e) = \begin{cases} 1 & \text{if } P(h) = 1 \quad (4.5) \\ \dfrac{P(h,e) - P(h)}{1 - P(h)} & \text{if } P(h) \neq 1 \wedge P(h|e) \geq P(h) \quad (4.6) \\ \dfrac{P(h,e) - P(h)}{P(h)} & \text{otherwise} \quad (4.7) \end{cases}$$

In a transactional databse, the implication of rule $X \Rightarrow Y$ presents the belief in hypothesis h of Y occurring in an transaction based on evidence e of this transaction containing X. Thus, with h and e observed in the database, we have $P(h|e) = conf(X \Rightarrow Y)$, $P(h) = P[Y]$ and $P[e] = P[X]$.

According to the above definition, the value of the CF measure is positive when $P(h|e) > P(h)$ whereas it is negative when $P(h|e) < P(h)$. In association rule mining, positive relations are usually considered. For example, in market basket problems, one may wish to find a positive "if *bread* then *milk*" relation so that the sale of *bread* can help boost the sale of *milk* [AS94]. The belief in hypothesis "sale of *milk*" should be increased based on the evidence "sale of *bread*". In other words, $P(milk|bread) > P(milk)$, or $CF(milk|bread) > 0$. Association rules are mined with lower bounds (i.e., minimum support and confidence thresholds) and only rules above these bounds are interesting. Thus, in this chapter we only consider the nonnegative CF shown in formula (4.6). The nonnegative CF observed in a transactional database is shown in formula (4.4).

In [BBSV02], a framework for assessing association rules was introduced with the concept of "very strong rule". An association rule $X \Rightarrow Y$ is very strong if both $X \Rightarrow Y$ and $\neg X \Rightarrow \neg Y$ are strong. The idea is that as the two rules are logical equivalence, finding evidence of both in data may help enforce the belief that the rule is important [BBSV02]. The CF measure used in this framework can help reduce the discovery of misleading rules, and improving the manageability and quality of the results.

4.1.5 Combination of Measures

As we have just reviewed, none of the interest, conviction, reliability or CF measure can be used to replace the support and confidence measures. Each of them has its own strong point in representing relations between items. They may be used to add more information to the support-confidence framework. Duanmu [Dua03] stated that three measures are needed to fully characterize association rules. The best combination is support, confidence and interest.

However, the combination approach may raise a new problem on how to use the different measures together, and also how to combine them. Moreover, some algorithms use only one measure to evaluate the data. The approach in [LMWY03], for example, uses confidence as the only measure to score the data.

4.1.6 Relationships Between Measures

Let's consider the difference between the confidence and CF measures of a rule $X \Rightarrow Y$:

$$conf(X \Rightarrow Y) - CF(X \Rightarrow Y) = conf(X \Rightarrow Y) - \frac{conf(X) - P[Y]}{1 - P[Y]} \quad (4.8)$$

$$= P[Y] \times \frac{1 - conf(X)}{1 - P[Y]} \quad (4.9)$$

We can imply that $conf(X \Rightarrow Y) > CF(X \Rightarrow Y)$. In addition, with small $P[Y]$, the difference between the confidence and CF measures will be small: $conf(X \Rightarrow Y) - CF(X \Rightarrow Y) < P[Y]$ with $P[Y] < conf(X \Rightarrow Y)$.

In formula (4.4), when $conf(X \Rightarrow Y)$ is replaced by $intr(X \Rightarrow Y) \times P[Y]$ from formula (4.1), we have:

$$CF(X \Rightarrow Y) = \frac{intr(X \Rightarrow Y) \times P[Y] - P[Y]}{1 - P[Y]} \quad (4.10)$$

$$= P[Y] \times \frac{intr(X \Rightarrow Y) - 1}{1 - P[Y]} \quad (4.11)$$

It means that the value of the CF measure is positive when its antecedence and consequence is positively correlated (i.e. $intr(X \Rightarrow Y) > 1$). The CF measure of a rule is zero when its antecedence and consequence have no correlation (i.e., $intr(X \Rightarrow Y) = 1$).

In formula (4.4), when $conf(X \Rightarrow Y) - P[Y]$ is replaced by $relb(X \Rightarrow Y)$ from formula (4.3), we have:

$$conf(X \Rightarrow Y) = \frac{conf(X) - P[Y]}{1 - P[Y]} \quad (4.12)$$

$$= \frac{relb(X \Rightarrow Y)}{1 - P[Y]} \quad (4.13)$$

The certainty factor also measures the variation of the probability that Y is in a transaction containing X like the reliability measure. However, the CF measure is "normalized" in the range of [-1,1]. For example, let's consider the case when $conf(X \Rightarrow Y) > P[Y]$. The reliability is $conf(X) - P[Y]$ which represents the increase of the probability of Y providing

X. The maximal increase is then $1 - P[Y]$. CF is the proportionate increase in the maximal possible increase.

4.2 Implication Relationships in Association Rules

In this section, we discuss the absolute and relative implications, and analyze the different measures especially the confidence and relative confidence (captured by CF) measures for association rule mining. The analysis is given based on a typical market basket problem, textual data and associative classification.

4.2.1 Absolute and Relative Implication

In Section 4.1, we have examined the problem of the support-confidence framework that although two rules R_1 ($university \Rightarrow study$) and R_2 ($university \Rightarrow data$) have the same values of support and confidence, they are at different levels of interestingness. The drawback of the confidence measure is that it is purely an absolute value of the implication of the rule "if X then Y". This implication is affected not only by the relation of X to Y, but also by the distribution of Y (the consequence of the rule). When the support value of Y is high, Y will likely be present in any transactions including those containing X regardless of how X is related to Y.

The confidence of a rule R reflects the implication "if a document contains X, then it will probably contain Y". The document will "contain Y" because X is related to Y so that once it mentions about X it is likely to mention about Y, or because Y is contained in some documents of the collection so that it is possibly contained in the document. Therefore, the confidence of R is determined by two factors: (i) the degree on how X is related to Y; and (ii) the density of random distribution of documents containing Y among documents containing X. The first factor can be considered as the actual value of the relation of X to Y, which we call it as relative confidence (or *R-confidence*) of rule R (denoted as $R\text{-}conf(R)$). As the set of documents containing Y is assumed to be distributed randomly, it applies the same to the degree of distribution of the documents containing Y on the whole collection of documents. The second factor, therefore, is represented by the support of Y (i.e., $supp(Y)$).

Let A be the set of documents containing X and B be the set of documents in A that contains Y, then $conf(R) = p(B|A)$ (probability of B on A). According to the two factors on the confidence of rule R, B would be compounded from (i) a set of documents B_1, which reflects the relations of X to Y; and (ii) a set of documents B_2, which reflects the random

distribution of the documents containing Y. Now, we can imply that $p(B_1|A) = R\text{-}conf(R)$, $p(B_2|A) = supp(Y)$ and $B = B_1 \cup B_2$. We then try to find out the relationship of the probabilities of these sets of documents. Here, the probabilities are considered in the sample space of A. It means that, for example, $p(B)$ denotes $p(B|A)$. That is,

$$p(B) = conf(R) \tag{4.14}$$

$$p(B_1) = R\text{-}conf(R) \tag{4.15}$$

$$p(B_2) = supp(Y) \tag{4.16}$$

From $B = B_1 \cup B_2$, we have

$$p(B) = p(B_1 \cup B_2) \tag{4.17}$$

$$= p(B_1) + p(B_2) - p(B_1 \cap B_2) \tag{4.18}$$

$$= p(B_1) + p(B_2) - p(B_1) \times p(B_2) \tag{4.19}$$

$$p(B) - p(B_2) = p(B_1) - p(B_1) \times p(B_2) \tag{4.20}$$

$$p(B_1) = \frac{P(B) - P(B_2)}{1 - P(B_2)} \tag{4.21}$$

Steps (4.18) and (4.19) are based on probability theorems of union of arbitrary events and multiplication rule for independent events [Szi93]. Note that two sets B_1 and B_2 are independent because B_2 is obtained from a random distribution. Now, we replace the probabilities of B, B_1 and B_2 from formulas (4.14), (4.15) and (4.16) respectively into formula (4.21) to derive the formula for $R\text{-}conf(R)$:

$$R\text{-}conf(R) = \frac{conf(R) - supp(Y)}{1 - supp(Y)} \tag{4.22}$$

Compared with formula (4.4), it is clear that $R\text{-}conf(R)$ is equivalent to the CF measure (i.e., $CF(R)$). In other words, we may say that the CF measure represents the relative implication relationship "if X then Y" without the effect from the distribution (or support) of consequence Y of the rule.

4.2.2 Analysis based on Market Basket Problem

Association rule mining is targeted for market basket problems which, for example, attempt to put *milk* near *bread* in a supermarket because eight out of ten (i.e., 80%) customers tend to pick up some *milk* after getting a loaf of *bread*. The manager of a supermarket then would

like to know how likely do the customers buy *bread* and then *milk* in order to organize the items. However, the implication *bread* $\Rightarrow$ *milk* may not reflect the actual relation of *bread* to *milk*. For example, only five out of ten (i.e. 50%) customers buy *milk* to drink with *bread* while the other three of the remaining five customers buy *milk* to have a cup of hot *milk* in cold winter. The five customers drinking *milk* with *bread* reflect the internal relationship of *bread* to *milk* - *bread* tastes good with *milk*. It may not change from winter to summer or from one area to another in a country. The three customers buying *milk* for drinking represents the distribution of *milk* (i.e., 60%) - it is not because they like to have *milk* with *bread*. When summer comes, the three customers may not like to drink "a cup of hot *milk*" anymore. For a group of the ten customers, who buy *bread*, the supermarket may not sell *milk* to the three customers who like to buy *milk* for drinking, but for the other five customers, which are reflected the relative implication of the CF measure (i.e., $50\% = \frac{80\%-60\%}{100-80\%}$), they remain to buy *milk* with *bread*. This measure represents the actual relation of *bread* to *milk*, which is not dependent on other factors of *bread* or *milk* such as the favor of a cup of hot *milk* in winter. The other factors, which only affect *bread* or *milk* separately, should be considered independently. For example, more *milk* should be provided in winter, or more *bread* should be stocked for supermarkets near a certain residence location.

Next, let's consider another example on association rule mining for the market basket problem. The process of mining association rules is originally stated by Agrawal *et al.* [AIS93] as a step towards enhancing databases with functionalities to process queries such as:

- "Find all the rules that have "Diet Coke" as consequence." These rules may help plan what the store should do to boost the sale of Diet Coke.
- "Find all the rules that have "bagels" in the antecedent." These rules may help determine what products may be affected, if the store discontinues selling bagels.

The first query finds rules in the form $X \Rightarrow$ *"Diet Coke"* for boosting the sale of Diet Coke. The items (or itemsets) X are found so that the sale of them help increase the sale of Diet Coke. However, a confidence value of, say, 50% of the rule *bagels* $\Rightarrow$ *"Diet Coke"* may not help determine if the sale of bagels will increase the sale of Diet Coke or not. Actually, in this case, if the usual sale of Diet Coke has the probability of less than 50%, then we can say that the sale of bread will increase the sale of Diet Coke and vice versa. Similarly, for the second query, the confidence measure in the rule *bagels* $\Rightarrow$ *"Diet Coke"* also unable to tell the impact of discontinuing the selling of bagels on *"Diet Coke"*.

Let's consider two sale items X and Y. The sale of Y can be specified by the probability of Y in a transaction. The sale of Y with the occurrence of X is $conf(X \Rightarrow Y)$. It means that the interest measure (i.e., $intr(X \Rightarrow Y) = conf(X \Rightarrow Y) - P[Y]$) can be used to represent the effect of X on Y. It can be used to measure how much the sale of bread can help boost the sale of Diet Coke or how much effect the discontinuance of the bagel sale can have on the sale of Diet Coke. However, as mentioned in Section 4.1.3, the pure difference in values between the confidence measure and consequence support of the rule does not truthfully reflect the implication as well as the correlation among the items. For example, if the probability of the sale of Diet Coke is 80%, then the reliability of the rule *bagels* $\Rightarrow$ *"Diet Coke"* is low regardless of how strong the effect of bagels has on *Diet Coke* (i.e., $relb(R) = conf(R) - P[Y] \leq 1 - P[Y]$). Here, the CF measure can be used to overcome the drawback of the reliability measure. The effect of X on Y now becomes proportionate effect.

4.2.3 Analysis based on Textual Data

In this experiment, we aim to investigate how good the different measures on association rules are in discovering the relations among the items contained in textual data using two document collections. The first document collection, namely D514, was described in Section 4.1. The document collection is obtained from published research papers on data mining with the intention that the rules found can be evaluated by meanings of the terms, which are familiar to the author. For example, we can recognize easily that *neural* $\Rightarrow$ *network* is a good rule, while *university* $\Rightarrow$ *data* is not.

The preprocessing of the D514 document collection consists of three steps. First, a stop-list of 571 common words [Fel98] is used to eliminate the stop-words from all documents. Next, a stemming process of suffix removal generates word stems. A total of 5,377 word stems are generated from the second step. Each word is considered as an item for association rule mining. There are 622 words of which document frequencies are equal to or greater than 10 (i.e., about 2% of the 514 documents). Some of these words are listed in Table 4.3 with their document frequencies.

The second document collection, namely *Reuters-R8*, contains 7,674 documents extracted from the popular Reuters-21578 dataset [DS04]. The Reuters-R8 dataset has been pre-processed by Cardoso-Cachopo [CC] using short-word (words with less than three characters) and stop-word removal, and stemming. Infrequent words with document frequency of

Table 4.3: Terms with document frequencies in the D514 document collection.

No	Term	Frequency
1	acquire	10
2	acquisition	10
3	action	10
4	adaptive	10
5	analytical	10
...	...	...
...	...	...
616	database	158
617	system	158
618	result	165
619	method	189
620	paper	247
621	mine	330
622	data	416

less than 100 were removed. There are 420 words in the dataset with document frequencies ranging from 100 to 3164. Some of these words are listed in Table 4.4 with their document frequencies.

Our experiment is conducted in the following steps according to a typical association rule mining process [AIS93, HP00]:

- Search for rules with the support value above a threshold value *minSupport.*
- Search for rules with the confidence value above a threshold value *minConfidence* among the rules found in Step 1.

The first step gathers those rules that are popular enough using *minSupport* of 1%. For convenience, we examine only single rules in the experiment. Single rules refer to rules of which the antecedence and consequence are single terms. With *minSupport* set to 1%, totals of 18,023 and 32,842 single rules are found in the first step from the D514 and Reuters-R8 document collections respectively. The second step is the main process of the experiments. In this step, the confidence, relative confidence (based on CF) and interest are used to mine interesting association rules. The rules found according to each measure will then be compared to evaluate the effectiveness of these measures.

The intention of the experiments is to compare the confidence, CF and interest measures in extracting interesting association rules from the document collection. Therefore, the best rules according to each measure are collected for comparison purpose. For the D514 document

Table 4.4: Terms with document frequencies in the Reuters-R8 document collection.

No	Term	Frequency
1	law	100
2	administr	100
3	juli	100
4	proce	100
5	commerci	100
...	...	...
...	...	...
415	year	2075
416	inc	2091
417	dlr	2462
418	for	2622
419	mln	2691
420	and	3164

Table 4.5: Association rules with highest values of confidence, CF (relative confidence) and interest generated from the D514 document collection.

Confidence	CF (relative confidence)	Interest
technology(32) ⇒ data(416)	neural(50) ⇒ network(72)	department(10) ⇒ university(11)
integrate(28) ⇒ data(416)	intelligence(15) ⇒ artificial(33)	play(13) ⇒ role(22)
deal(26) ⇒ data(416)	play(13) ⇒ role(22)	care(13) ⇒ health(15)
extraction(26) ⇒ data(416)	execution(11) ⇒ time(78)	text(12) ⇒ document(17)
aim(24) ⇒ data(416)	cdna(10) ⇒ gene(35)	commonly(10) ⇒ quantitative(16)
environment(24) ⇒ data(416)	lack(12) ⇒ analysis(149)	cdna(10) ⇒ express(20)
solve(24) ⇒ data(416)	apriori(11) ⇒ algorithm(157)	bound(10) ⇒ low(21)
mine(330) ⇒ data(416)	massive(12) ⇒ method(189)	intelligence(15) ⇒ artificial(33)
event(23) ⇒ data(416)	decomposition(10) ⇒ model(140)	text(12) ⇒ retrieval(18)
handle(22) ⇒ data(416)	automatic(20) ⇒ mine(330)	cdna(10) ⇒ gene(35)

collection, there are 62 rules that have the CF value of 100%. The confidence value of them will also be 100% according to formula (4.4). Thus, these rules are not meaningful for comparison between the two measures. It is also fair to discard these best rules for comparison with the interest measure. Table 4.5 lists the ten highest rules for each measure without counting the 62 best rules generated from the D514 document collection. The number given in the parentheses next to a term is the document frequency of the term.

As shown in Table 4.5, all the ten rules of highest confidence are in the form of $X \Rightarrow data$ which are not very interesting. As we have discussed earlier that the confidence measure of a rule is influenced by two factors, namely the CF and the consequence support (P[Y]). In this case, due to the very high value of support of the term *data* (416/514=81%), all the rules in the form of $X \Rightarrow data$ will potentially have a high value of confidence. In fact, amongst the

Table 4.6: Two rules with different values of interest.

Rule	P[X]	P[Y]	P[X∧Y]	Conf(R)	intr(R)
R_1	2%	2%	1%	50%	25
R_2	4%	4%	2%	50%	12.5

100 best rules with the confidence measure, there are up to 95 rules in which the consequence is *data* or *mine*, the two most frequent terms in the document collection. In contrast, the rules mined with the CF measure are quite interesting in the data mining domain. Having the noise on the frequency of items removed, the terms of the rules with high CF are strongly related.

In Table 4.5, the ten rules with the highest interest values consist of strongly correlated terms. However, they are not so interesting in the domain of data mining. For example, we can see that there are five rules found with the CF measure (i.e., *neural* $\Rightarrow$ *network*, *intelligence* $\Rightarrow$ *artificial*, *apriori* $\Rightarrow$ *algorithm*, *decomposition* $\Rightarrow$ *model* and *automatic* $\Rightarrow$ *mine*) which are related to one of the data mining topics. But only two rules (i.e. *intelligence* $\Rightarrow$ *artificial* and *text* $\Rightarrow$ *retrieval*) are found with the interest measure. We can also observe that the term frequencies of the rules are relatively small. The reason is that with a given confidence value of a rule $X \Rightarrow Y$ (or $Y \Rightarrow X$), according to formula (4.1), the interest value is inversely proportional to the value of P[Y] (or P[X]). It means that infrequent itemsets are preferable for the interest measure. For example, Table 4.6 shows two rules R_1 and R_2 formed from itemsets X and Y that occur together in half of the documents containing them (i.e., $P[X \wedge Y] = \frac{1}{2}P[X] = \frac{1}{2}P[Y]$). The two rules then have the same value of confidence of 50%, but the interest measure of R_1 (i.e., 25) is much higher than R_2 (i.e., 12.5) as R_1 contains itemsets with lower frequency in comparison with R_2.

We have also used association rules to find related terms for certain topics from the D514 document collection. This is done as follows. We use three search terms, namely "retrieval", "association" and "web", which are related to one of our research topics called "Web information retrieval using association rules", to find related terms from the document collection using different measures. Five related terms which are most related to each search term based on different measures are collected. For example, if the search term is "retrieval", the five terms T with maximal values for the confidence measure such that $conf(T \Rightarrow retrieval)$ or $conf(retrieval \Rightarrow T)$ are selected. Similarly, the top five related terms for the relative

Table 4.7: The top five related terms from search terms.

Search term	CF	Freq	Confidence	Freq	Interest	Freq
retrieval	information	147	information	147	text	12
	text	12	data	416	document	17
	process	127	mine	330	index	21
	application	135	process	127	web	30
	document	17	application	135	term	30
association	rule	101	data	416	apriori	11
	apriori	11	rule	101	item	12
	item	12	apriori	11	transaction	16
	database	158	database	158	prune	11
	transaction	16	item	12	rule	101
web	internet	16	data	416	internet	16
	information	147	mine	330	site	14
	site	14	paper	247	interface	12
	interface	12	information	147	text	12
	text	12	internet	16	document	17

confidence (or CF) and interest measures are also collected. The related terms found are listed in Table 4.7 with their frequencies.

We can observe that terms that are selected by the confidence measure have very high frequencies while the interest measure mainly focuses on terms with very low frequencies. For instance, among the terms related to *retrieval*, the term *mine* has a very high confidence with *retrieval* (the third highest one). This is not because the two terms have a close relation but simply the term *mine* has a very high document frequency. In contrast, the interest measure eliminates quite frequent phrases such as *information retrieval*, *retrieval process* and *retrieval application* as found under the CF measure to reserve room for rather infrequent but not very related terms of *index* and *term*.

In addition, association rules are also generated from the Reuters-R8 document collection according to each measure. Table 4.6 lists the ten highest rules for each measure. The number given in parentheses next to a term is the document frequency of the term.

The Reuters-R8 document collection was extracted from the Reuters-21578 dataset which includes documents appeared in Reuters newswire in 1987 [DS04]. This collection of documents consists of many financial reports containing income statements or balance sheets. As data from these reports are often presented in a table format, abbreviations are frequently used. For example, the abbreviations which are commonly used in financial reports include avg (i.e. average), shr (i.e. share), ct (i.e. cent), net (i.e. net profit/loss), rev (i.e. revenue),

Table 4.8: Association rules with highest values of confidence, CF (relative confidence) and interest generated from the Reuters-R8 document collection.

Confidence	CF (relative confidence)	Interest
avg (407) ⇒ net (1818)	avg (407) ⇒ net (1818)	intent (115) ⇒ letter (146)
avg (407) ⇒ shr (1498)	avg (407) ⇒ shr (1498)	confer (119) ⇒ new (142)
mth (317) ⇒ net (1818)	mth (317) ⇒ net (1818)	chief (145) ⇒ execut (174)
competit (112) ⇒ and (3164)	mth (317) ⇒ shr (1498)	japanes (154) ⇒ japan (241)
rev (1001) ⇒ net (1818)	rev (1001) ⇒ net (1818)	european (110) ⇒ commun (168)
mth (317) ⇒ shr (1498)	competit (112) ⇒ and (3164)	canadian (126) ⇒ canada (171)
rev (1001) ⇒ shr (1498)	rev (1001) ⇒ shr (1498)	chief (145) ⇒ offic (187)
countri (288) ⇒ and (3164)	qtly (393) ⇒ ct (2069)	intent (115) ⇒ sign (205)
economi (111) ⇒ and (3164)	qtly (393) ⇒ record (834)	currenc (148) ⇒ dollar (211)
put (107) ⇒ and (3164)	barrel (159) ⇒ oil (434)	economi (111) ⇒ econom (208)

mth (i.e. month), qtly (i.e. quarterly), etc. The use of abbreviations makes these terms strongly related. The best rules found with the confidence and CF measures shown in Table 4.8 also support this observation. However, we can also see that the common conjunction "and", which is the most frequent word in the document collection, occurs four times in the best rules found with the confidence measure. This shows that the confidence measure is biased towards frequent terms. Similar to the D514 document collection, we also observe that the ten rules with the highest interest values found with the Reuters-R8 document collection shown in Table 4.8 consist of strongly correlated terms presenting common English phrases with relatively small term frequencies. Here, as mentioned in Section 4.1.1, frequent items would never form a rule with high interest value. It means that the use of interest may never result with frequent terms which could be relevant to the document collection.

4.2.4 Analysis based on Associative Classification

Associative classification (AC) [LHM98, LHP01, LMW01] uses association rules for data classification. This approach is based on the implication relationships from itemsets to classes of class association rules [LHM98], which are represented by the confidence measure for data classification. In this experiment, apart from using confidence, we have also used the realtive confidence (or CF) measure for AC. The performance of associative classification using the confidence and CF measures is then used to evaluate the effectiveness of the measures in extracting the implication relationships among items for associative classification.

Generally, associative classification consists of three major processes [LHM98, LHP01, LMW01], namely rule discovery, rule selection and classification, as discussed in Section 2.3. To classify a test sample d, class association rule $R : iset \Rightarrow c$, where $iset$ is an itemset and c

is a class, with the highest confidence value that matches d is selected. The test sample d is then assigned to class c of R [LHM98]. The idea is that the rule with the highest confidence value should give the best accuracy for class prediction. However, as some discovered rules are removed during the rule selection process, the rule used for classification of a data sample may not be the best rule according to the confidence measure. The purpose of the rule selection process is to optimize classification accuracy. It simulates a classification process on the training dataset and selects the subset of rules which gives the best accuracy. The rationale behind is that, due to interactions among the rules, the best rules may not form the best set of rules for classification. Thus, the rule selection process is necessary for improving the accuracy of AC.

In fact, the set of rules discovered in the rule discovery process can be used directly for the classification process. In this case, the best rule for each data sample according to the confidence measure can always be obtained. Without the rule selection process, AC may not be as accurate as that using all three AC processes. However, it can reflect truthfully how good the set of rules discovered with the support and confidence measures is for classification.

In this experiment, we have implemented two associative classification algorithms, namely CBA [LHM98] which is the conventional associative classification approach, and msCBA [LMW01] which is an improved version of CBA with the use of multiple class supports to enhance classification accuracy. For comparison between the confidence and CF measures for AC, we have implemented four different approaches based on the CBA and msCBA algorithms. They are denoted as CBA(conf), msCBA(conf), CBA(CF) and msCBA(CF), which use the confidence and CF measures respectively. To evaluate the effectiveness of the measures on classification accuracy, we have implemented these four approaches with and without the rule selection process. The implementation without the rule selection process is useful for comparing directly the set of rules discovered from the different measures for the classification process, whereas implementation with the rule selection process helps evaluate the effectiveness of the measures on the performance of the overall AC.

The evaluation is based on the *accuracy* of the classification. In addition, we also use the standard *F-measure* which is based on *precision* and *recall* to evaluate the classification performance for each class [BYRN99]. The accuracy of AC, and the precision, recall and F-measure for a class C are defined as follows:

$$accuracy = \frac{total\ number\ of\ data\ samples\ classified\ correctly}{total\ number\ of\ data\ samples\ classified} \tag{4.23}$$

Table 4.9: Adult Dataset.

Class	No. of records	Training set	Test set
>50K	37,155	24,720	12,435
≤50K	11,787	7,841	3,846
Total	48,842	32,561	16,281

Table 4.10: Reuters-R8 Dataset.

Class	No. of documents	Training set	Test set
acq	2,292	1,596	696
crude	374	253	121
earn	3,923	2,840	1,083
grain	51	41	10
interest	271	190	81
money-fx	293	206	87
ship	144	108	36
trade	326	251	75
Total	7,674	5,485	2,189

$$precision(C) = \frac{total\ number\ of\ data\ samples\ classified\ correctly\ into\ class\ C}{total\ number\ of\ data\ samples\ classified\ into\ class\ C} \quad (4.24)$$

$$recall(C) = \frac{total\ number\ of\ data\ samples\ classified\ correctly\ into\ class\ C}{total\ number\ of\ data\ samples\ belonging\ to\ class\ C} \quad (4.25)$$

$$F\text{-}measure(C) = \frac{2 \times precision(C) \times recall(C)}{precision(C) + recall(C)} \quad (4.26)$$

Two datasets with uneven class distribution have been used in the experiment. The first dataset is the Adult (also called Census) dataset obtained from the UCI Machine Learning Repository [NHBM98]. The second dataset is the Reuters-R8 dataset which has been discussed in Section 4.2.3. This dataset was formed from 8 classes of the Reuters-21578 dataset [DS04] that have the most number of positive training samples. Table 4.9 and Table 4.10 list the properties of the Adult and Reuters-R8 datasets which have already been separated into training and test sets.

The experiments have been conducted with different support threshold values. For the Adult dataset, the minimum support for CBA and total minimum support for msCBA are ranged from 0.2% to 1% and from 1% to 5% respectively, whereas for the Reuters-R8 dataset, the minimum support for CBA and total minimum support for msCBA are ranged from 0.6%

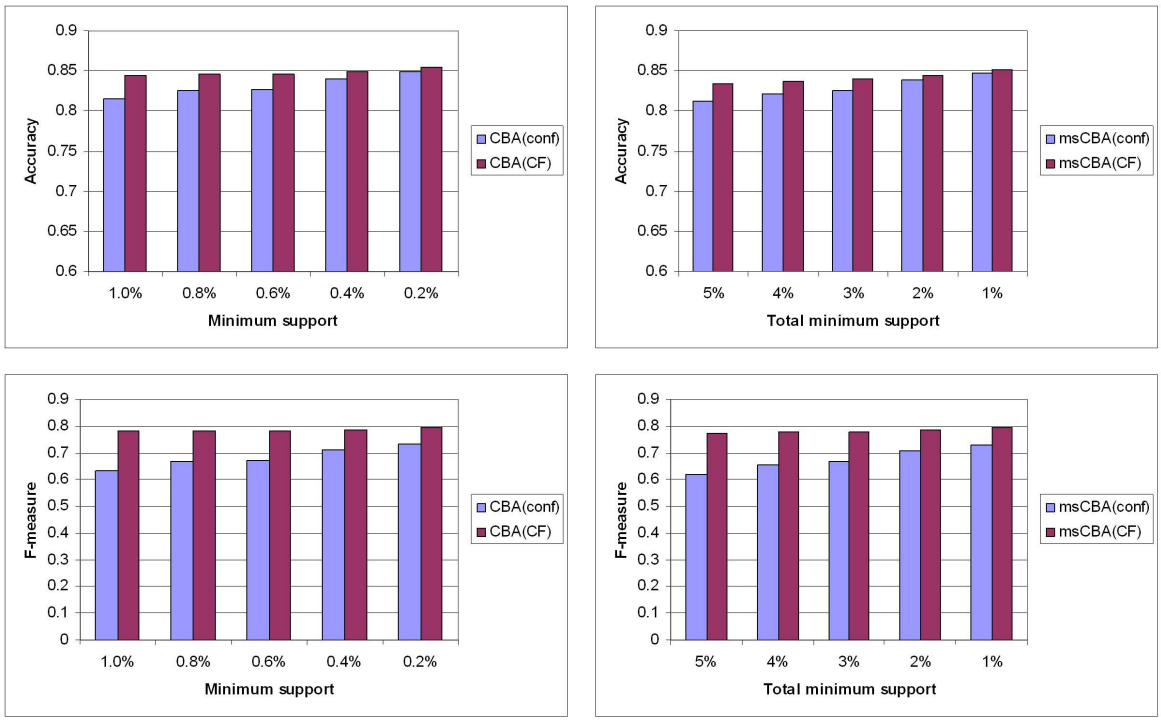


Figure 4.1: Performance results of AC approaches without the rule selection process based on the Adult dataset.

to 1.2% and from 8% to 16% respectively. The minimum confidence and CF thresholds were both set to 50%. Figure 4.1 and Figure 4.2 show the performance results of the algorithms implemented without the rule selection process based on the Adult dataset and Reuters-R8 dataset respectively. The performance results have shown that the associative classification approaches using the CF measure have achieved better accuracy and F-measure in comparison with those using the confidence measure. The improvements in accuracy and F-measure for the Adult dataset are 1.86% and 3.78% respectively, whereas for that of the Reuters-R8 dataset, the improvements are 1.45% and 10.06% respectively.

Let's consider the performance results of AC approaches without the rule selection process based on the Adult dataset with support threshold of 1%, which is given in Table 4.11. This attribute dataset consists of transactions of employee information. Each employee belongs to one of the following classes: "$\leq$50K" or ">50K" according to the *income* attribute. In the training set, there are 76% of the transactions belonging to the class "$\leq$50K", and the remaining 24% belong to the class ">50K".

The F-measure values for the two classes have shown that the associative classification approaches using the CF measure have achieved a more balanced performance on these two

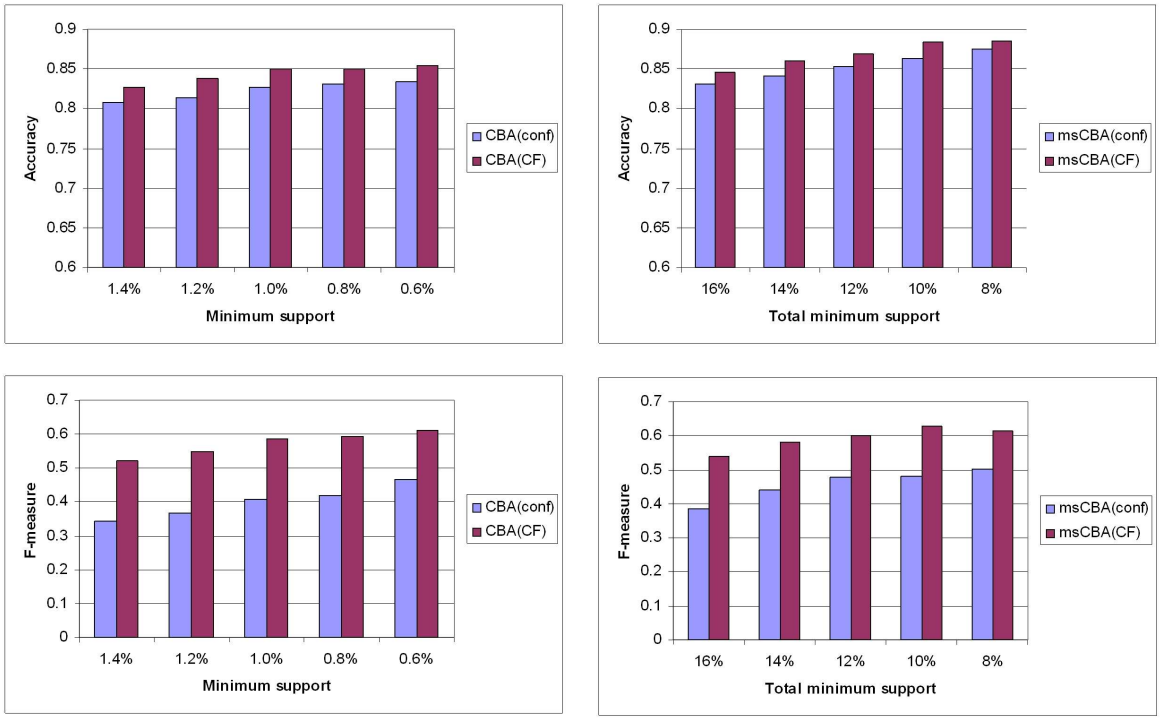


Figure 4.2: Performance results of AC approaches without the rule selection process based on the Reuters-R8 dataset.

classes. In contrast, the AC approaches with the confidence measure have performed poorly on the infrequent class of ">50K". The reason is that the two classes of "≤50K" and ">50K" are distributed very unevenly, and the rule discovery process with the confidence measure is biased towards the rules of the more frequent class of "≤50K". This can be explained as follows. For example, let's examine two rules ($X \Rightarrow$ "≤50K") and ($X \Rightarrow$ ">50K") with a confidence value of 80%. According to formula (4.4), the corresponding CF values of these two rules will be 17% and 73%. In fact, as 76% of the transactions belong to the class "≤50K", the confidence of the rule ($X' \Rightarrow$ "≤50K") with a random itemset X' is about 76%. As such, the confidence value of 80% does not show strong implication from X to "≤50K". The weak implication is also suggested by the value of the CF of 17%. In contrast, the rules in the form of ($X \Rightarrow$ ">50K") with the same 80% confidence value have shown very strong implication because only 24% of the transactions belong to the class ">50K". This is also shown by the value of the CF of 73%. Therefore, the association rules that are discovered using the confidence measure possibly contain many weak rules of the class "≤50K", and too few rules of the class ">50K".

Table 4.11: Performance results of AC approaches without the rule selection process based on the Adult dataset with support threshold value of 1%.

Approach	Accuracy	F-measure		
		"≤50K"	">50K"	Average
CBA(conf)	81.5%	89.1%	37.2%	63.2%
CBA(CF)	84.5%	89.9%	66.1%	78.0%
msCBA(conf)	84.7%	90.8%	55.3%	73.0%
msCBA(CF)	85.2%	90.3%	68.8%	79.6%

Table 4.12: Contingency table of AC approaches based on the Adult dataset.

Approach	Actual class	No. of data samples classified into "≤50K"	No. of data samples classified into ">50K"
CBA(conf)	"≤50K"	12375	60
	">50K"	2953	893
CBA(CF)	"≤50K"	11279	1156
	">50K"	1375	2471
msCBA(conf)	"≤50K"	1170	735
	">50K"	1439	2407
msCBA(CF)	"≤50K"	11220	1215
	">50K"	1191	2655

The discrimination against rules of the infrequent class can also be observed using the contingency table obtained from the classification results given in Table 4.12. As shown from this table, many data samples of the infrequent class are mis-classified into the frequent class while much fewer data samples of the frequent class are mis-classified into the infrequent one with the AC approaches using the confidence measure. For example in the CBA(conf) approach, there are 2,953 data samples of class ">50" which are mis-classified while only 60 data samples in "≤50" are mis-classified. In contrast, the AC approaches using the CF measure give a more balanced classification among the classes. The numbers of data samples mis-classified between the two classes (1,375 vs. 1,156 for CBA(CF) and 1,191 vs. 1,215 for msCBA(CF)) are similar. It means that the "implication strength" among the rules between the two classes discovered and used for classification using the CF measure is similar.

Table 4.13 shows the average performance of the AC approaches with the rule selection process (i.e., the complete CBA and msCBA processes). We can see that the use of the rule selection process has improved the performance for AC approaches. Nevertheless, the AC

Table 4.13: Average performance of AC approaches with the rule selection process.

Approach	Adult		Reuters-R8	
	Accuracy	F-measure	Accuracy	F-measure
CBA(conf)	84.83%	0.76	86.87%	0.54
CBA(CF)	85.64%	0.78	88.39%	0.58
msCBA(conf)	84.38%	0.75	85.50%	0.58
msCBA(CF)	85.37%	0.77	87.70%	0.61

approaches with the CF measure still have better performance than those with the confidence measure in terms of accuracy and F-measure.

4.3 Associative Feature Selection for Text Mining

Text mining applications always need to deal with large and complex datasets of textual documents that contain much irrelevant and noisy information. Feature selection aims to remove the irrelevant and noisy information by focusing only on relevant and informative data for use in text mining. Feature selection can be supervised with human support in labelling the data, or be unsupervised without any human involvement.

In this section, we propose an unsupervised feature selection approach for text mining called *associative feature selection*. The approach is based on the assumption that relevant features in a textual document collection are closely associated. The highly associated features can be searched using association rule mining [AIS93, Heg03, HP00]. In addition, we evaluate the performance of our proposed associative feature selection approach and compare its performance with other unsupervised feature selection approaches that are based on document frequency and term strength.

4.3.1 Feature Selection Approaches for Text Mining

Feature selection is used to select a "better" subset of features that can describe data from an original dataset. Feature selection aims to (i) focus on the relevant data; and (ii) reduce the amount of data [LMY03]. Feature selection approaches are based on either exhaustive or heuristic search. Exhaustive feature selection approaches search for all possible combinations of features and find an optimal one based on an evaluation criterion. Let N denote the number of features in the original dataset. The total number of candidate subsets is 2^N. Although it is not always necessary to scan all possible subsets, exhaustive search is still quite

computationally expensive [KS96]. Heuristic feature selection approaches employ heuristics in conducting search. One approach of heuristic search is to score features based on some heuristic measures. The score of a feature indicates how relevant it is to the dataset. The selection process is then very straightforward. Features will be selected if their scores are above a predefined threshold.

The "bag-of-words" approach [LMY03] is commonly used to analyze textual documents. In this approach, a document is considered as a set of words or phrases called *terms*. When applying data mining techniques to textual documents, a document is considered as an instance (data sample), while terms (words or phrases) are considered as *features* (or items). There are a number of feature selection approaches, which can be applied effectively to textual data. Most of them are based on a scoring scheme of terms [Mla98]. The scores of features represent the quality the terms in the document collection. A term with a high score means it is important or relevant to the dataset. In supervised approaches, term scores are based on a labelled training set, which contains class information. Some of the popular supervised feature selection approaches include information gain (IG), mutual information (MI) and χ^2 statistics (CHI) [YP97].

Unsupervised feature selection approaches are based on heuristics for estimating the quality of terms in a dataset. For a dataset of textual documents, the heuristics generally focus on term distribution among the documents in the dataset. The popular unsupervised feature selection approaches include document frequency (DF) and term strength (TS). Document frequency is a simple but effective measure for feature selection. In [YP97], Yang *et al.* concluded that DF is among the best measures (as good as IG and CHI) for selecting informative features. Document frequency of a term is the number of documents in which the term occurs. The feature selection approach calculates the document frequency for every term and removes those terms whose document frequency is less than a predefined threshold. The basic assumption is that frequent terms are more important and relevant to the dataset in comparison with the infrequent ones.

Term strength was proposed in [WS92] originally for stop-word removal. This approach estimates the strength of a term based on how likely it appears in "closely-related" documents. It is based on the heuristic that documents with many shared words are related, and that terms occurring in heavily overlapping areas of related documents are relatively informative [YP97]. The approach consists of the following two steps:

- *Find pairs of similar documents.* This step calculates the similarities between all pairs of documents in the dataset, $sim(d_i, d_j)$, using the cosine value of the two document vectors. Two documents d_i and d_j are then considered "similar" if $sim(d_i, d_j)$ is above a predefined threshold ξ.

- *Calculate term strength.* The strength of a term $s(t)$ is computed based on the estimated conditional probability that the term t occurs in a document d_i when it occurs in document d_j, which is similar to d_i: $s(t) = p(t \in d_i \mid t \in d_j \wedge sim(d_i, d_j) \geq \xi)$.

Unsupervised feature selection approaches can save the cost of labelling the data and avoid the problem on inaccuracy of homogeneity between the training and test datasets in the supervised process [LMY03]. This characteristic is especially important for text mining in which we always need to deal with a huge amount of documents on various topics.

4.3.2 Associative Feature Selection Approach

In the previous sections, we have shown how terms could be scored based on their distribution on a document collection. The distribution of a term could be examined independently like DF or in relation with other terms like TS. In this section, we propose an approach to select relevant terms based on the associations among them. Such associations can be discovered using association rule mining.

Generally, a document of a dataset belongs to one or more *topics* in a certain field or area. The topics of all documents in the dataset form their theme called *domain*. In short, a domain can contain multiple topics, and each topic is discussed in some documents. A term that is relevant to the dataset means it is relevant to the domain of the dataset. A relevant term should then be relevant to some topics of the domain. It may be used to explain or illustrate the topics or some concepts of the topics. As terms relevant to a topic will possibly occur in documents belonging to the topic, a relevant term is likely to occur with some other relevant terms (which are related to the same topics). The remaining irrelevant terms, which are not related to any topics, are probably distributed randomly. The probability of these terms to occur in a document does not depend on the topics of the document. This is illustrated in Figure 4.3. The associative features of relevant terms and irrelevant terms from documents in a domain can be summarized as follows:

- A relevant term is probably associated with other relevant terms.

- An irrelevant term is not likely to be associated with other terms.

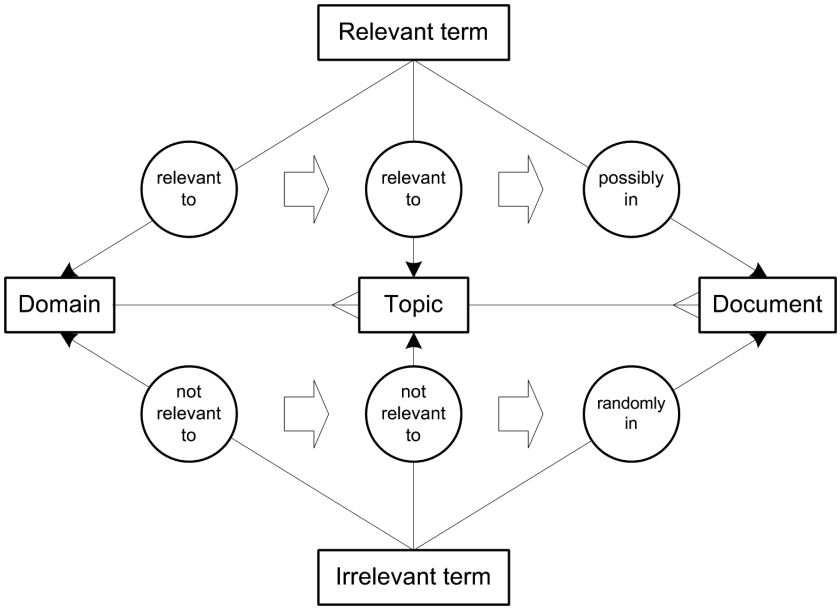


Figure 4.3: Relations of relevant and irrelevant terms in a domain.

The proposed associative feature selection approach is based on the heuristics discussed above to separate relevant and irrelevant terms. Terms that occurred in many association rules means that they are associated with many other terms. These terms should then be assigned with a high score so that they are considered as relevant terms. On the contrary, terms that occur infrequently in association rules should be assigned with a low score for irrelevant terms. The assignment of scores to features comprises the following three steps: (1) determine the constraints of the association rules; (2) search for association rules satisfying the constraints; and (3) score the features based on the association rules found.

In the first step, we determine the constraint F for the association rules such that $F : ARs \Rightarrow Boolean$. Here, ARs is the set of all association rules on the set of terms. Conventionally, the constraint for association rules is that support and confidence are greater than the threshold values of *minSupport* and *minConfidence*. The constraint F can also include other measures for mining association rules. In our implementation, the CF measure is used to improve the quality of the selected terms.

The second step searches for association rules that satisfy constraint F. The typical approach for searching association rules is the classical Apriori algorithm given in [AS94]. In this approach, the support measure is first used to filter out most of the rules that do not satisfy *minSupport*, the confidence measure (or other measures) is then applied to the remaining rules that also satisfy *minConfidence*.

Finally, terms are scored based on the association rules found in the second step. A term will have a high score if it occurs in many rules. Currently, we simply use the number of rules in which a term occurs as the score of the term. The calculation of a term score can also be based on the "quality" of the rules (for example, support and confidence), position of the term (antecedence or consequence) and the scores of other terms in the set of rules it occurs.

4.3.3 Performance Analysis

We have conducted an experiment to measure the performance of the proposed associative feature selection approach using the D514 document collection. The document collection was obtained from published research papers on data mining with the intention that the terms found under the measures could be evaluated by their meanings that are familiar to the author. The preprocessing of the document collection has been described in Section 4.3.1.

We have implemented the associative feature selection approach and the other two approaches that are based on document frequency (DF) and term strength (TS) for comparison. All approaches are based on a term scoring scheme. The DF approach scores terms based on frequency. The frequency of a term is the number of documents containing the term. The TS approach scores terms based on their strength [WS92]. It first calculates the similarities $sim(d, e)$ of all pairs of documents in the document collection. Pairs of documents are then selected if these similarities are above a predefined threshold. Finally, the strength of terms is estimated based on these pairs of documents. In associative feature selection, we use the scoring procedure discussed in Section 4.4.2 to calculate the scores. There are two implementations: one uses the confidence measure (called AF-C) and the other uses the CF measure (called AF-CF). The threshold values for support and confidence were set to 2% and 30% respectively.

Selection of terms after scoring is straightforward. A threshold is then defined and the features whose score above the threshold value are selected. The threshold can be determined by a statistical approach as in [WS92]. The number of terms can be altered by increasing or decreasing the threshold value.

Measuring the Goodness of Features

The implementation of different approaches will generate different sets of relevant and irrelevant terms from the total of 622 terms selected. Therefore, a standard labelling of terms

(relevant or irrelevant) is needed so that we can evaluate the goodness of each feature selection approach. In this experiment, the labelling is done in two steps. In the first step, terms are manually classified into five groups as follows:

1. Topics, tasks, approaches, applications: e.g., association, classify, cluster.
2. Concepts, terms: e.g., term, pattern, set, database, text, algorithm, etc. They are concepts used more frequently in data mining than other IT topics.
3. Words specially used for a topic of data mining: e.g., frequency, itemset, apriori (association rule mining); gene, tissue (data mining for biology); attribute, dimension (database mining); sequence, parallel, regression (data mining approaches), etc.
4. Words that are also popular in other IT topics: e.g., system, approach, software, etc.
5. Common words: e.g., show, define, increase, analyze, accurate, automatic, intelligent, challenge, etc.

The second step simply groups the first three groups into a set labelled as relevant terms. This set consists of a total of 118 terms. The terms in groups 4 and 5 which are also used in documents in the domains other than "data mining" are labelled as irrelevant. Here, the classification of terms into the above five groups makes the "manual labelling" process easier and more accurate. It also helps justify if an item is relevant or not. The performance is evaluated using the standard *precision* and *recall* measures [BYRN99] based on the document collection. The *precision* and *recall* measures for feature selection are defined as follows:

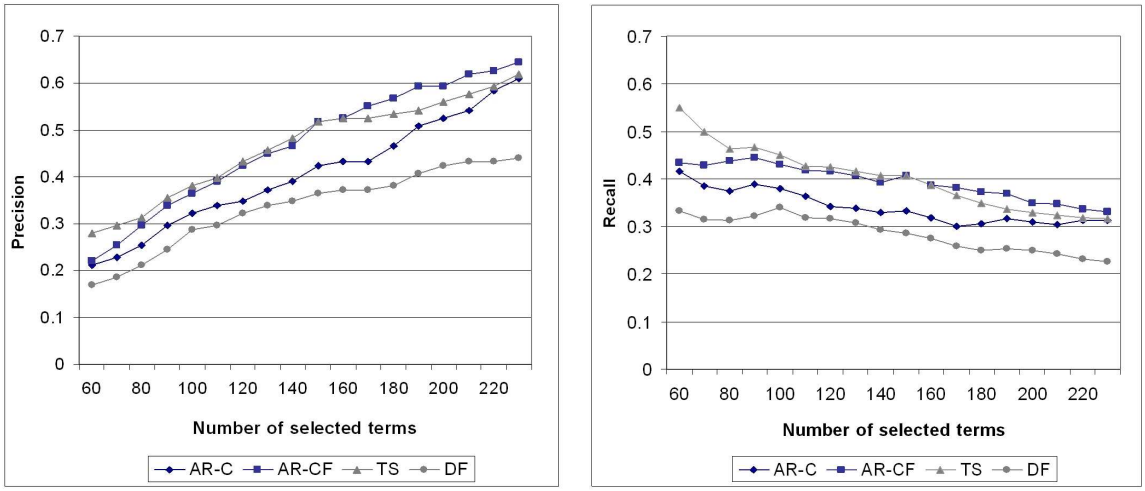
$$precision(C) = \frac{number\ of\ terms\ selected\ and\ relevant}{number\ of\ terms\ selected} \tag{4.27}$$

$$recall(C) = \frac{number\ of\ terms\ selected\ and\ relevant}{number\ of\ terms\ relevant} \tag{4.28}$$

Experimental Results

The results of the feature selection approaches are lists of terms with the corresponding scores. To select k relevant terms, we simply sort the list of terms based on the scores and obtain the k terms with the highest scores. To evaluate the goodness of each approach, the k relevant terms are selected and compared to the standard labelled set using the precision and recall measures.

We have conducted our experiment with a wide range of values on k for each feature selection approach. The range of k was set from 59 (a half of the number of relevant terms of



(a) Precision (b) Recall

Figure 4.4: Performance results of different feature selection approaches.

118) to 236 (two times of 118). The results on precision and recall are shown in Figure 4.4(a) and Figure 4.4(b) respectively. The experimental results have shown that the associative feature selection approach performs better than the approach using document frequency and as good as when the term strength is used. It can be explained that document frequency while focusing only on frequent terms may miss some important but rare ones. Association rules (AR) weighting is similar to term strength (TS) in considering the relations among terms. These two approaches are based on the heuristic that relevant terms on a textual document collection should have some relations regarding the distribution among the dataset.

The experimental results have also shown that the CF measure is appropriate for discovering good association rules. The precision and recall have increased considerably when the CF measure (i.e., the AR-CF approach) is used instead of the confidence measure (i.e., the AR-C approach).

The unsupervised feature selection approaches using TS and AR exploit the co-occurrences of terms on textual documents. However, as the approach using TS requires to compute the similarities between all pairs of documents in the dataset, its time complexity would be $O(N^2)$ where N is the number of documents. This makes the feature selection approach using TS a weak choice for large datasets of documents. The runtime performance of the approach based on AR depends on the process of searching for association rules. And some AR mining approaches have good runtime performance [BYRN99], which is directly proportional to the size of the dataset (i.e. the time complexity would be $O(N)$).

4.4 Summary

In this chapter, we have studied a framework for assessing association rules using the CF measure. While the confidence measure is appropriate for assessing absolute value of the implication, the CF measure can reflect truthfully the relations of items by removing the noisy substance of frequency from the confidence measure. We have shown that the CF measure is effective over the confidence measure, especially when the dataset contains items with uneven frequency distributions such as the textual dataset. We have also shown that the CF measure can also be used to improve associative classification for datasets with classes having uneven frequency distribution.

In this chapter, we have also proposed an unsupervised associative feature selection approach for text mining. Based on association rule mining, the proposed approach discovers the associative strength of terms among documents. This hidden knowledge is then used to score the terms according to their importance in the dataset. The performance results of the associative feature selection approach have shown that the use of CF has performed better than that of confidence for associative feature selection from textual documents. However, the proposed associative feature selection approach using association rule mining is only evaluated based on the criterion of how relevant the selected features are to the document dataset. To have a more accurate evaluation of the approach, further research should be performed on its effectiveness to the later data mining process such as categorization. For example, Yang et al. [9] have examined different feature selection approaches for text categorization. Several feature selection approaches are implemented and the features selected by different approaches are then used for categorization purposes. The evaluation of the categorization process could then be used to measure the effectiveness of different feature selection approaches for text mining.

Chapter 5

Multiple-step Rule Discovery for Associative Classification

Associative classification (AC) [LHM98, LHP01] takes advantage of association rule mining [AS94, HPY00] in the rule discovery process to extract high quality rules that can accurately generalize the training dataset. This approach has achieved high accuracy in comparison with other classification approaches [LHM98, LHP01, LMW01, AZ02]. However, one of the main problems of using association rule mining in AC is its very large search space of possible rules. This may cause performance degradation in the rule discovery process, especially with small support threshold values which are very important for building accurate associative classifiers from large training datasets. In other words, the rule discovery process may be very computational expensive in order to achieve accurate associative classification from large training datasets. In this chapter, we propose the Multiple-step rule discovery approach for AC called Mstep-AC, which aims to achieve better associative classification accuracy while maintaining the rule discovery process to the level of complexity comparable with conventional AC approaches.

The rest of this chapter is organized as follows. In the next section, we discuss the class association rule discovery process for AC including the tradeoff between classification accuracy and the complexity in rule discovery of conventional AC approaches from large training datasets. In Section 5.2, the proposed Mstep-AC approach is discussed. The performance evaluation is then given in Section 5.3. Finally, we give a summary of the chapter in Section 5.4.

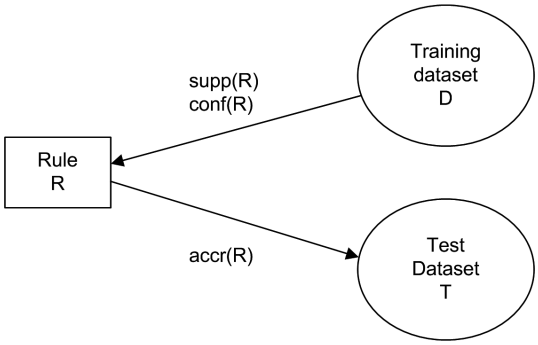


Figure 5.1: Support and confidence in rule prediction.

5.1 Class Association Rule Discovery

This section discusses how the parameters such as the support and confidence thresholds affect the performance of the rule discovery process and how this process affects the overall performance of associative classification in terms of both accuracy and efficiency (i.e., computational complexity).

5.1.1 Support and Confidence

Associative classification is based on the "implication" relationships between data items and class labels for class prediction. This relationship is represented in class association rules (CARs) in the form of "if *iset* then c" (i.e., $iset \Rightarrow c$) which enables the prediction of the class label of a data sample from the items it contains. Let $R : iset \Rightarrow c$ be a CAR and D be the training dataset. The confidence value of R (i.e., $conf(R)$) is the percentage of data samples that contain class label c among the data samples containing itemset *iset* in D. Thus, the confidence value of R is the accuracy of the prediction "if *iset* then c" among the data samples in D. The support value is the percentage of data samples containing *iset* and c with respect to the number of all data samples in D. It shows the proportion of data samples in D in which the rule occurs.

Let's consider the prediction accuracy of R (i.e., $accr(R)$) as the accuracy when R is used for class prediction on a test data sample. Figure 5.1 shows the roles of support and confidence in rule prediction. A class association rule R is generated from the training dataset D and then used to classify data samples from test dataset T. The class association rule is generated based on the support and confidence measures. The goodness of the classification is represented by the prediction accuracy measure.

Let's consider the problem formally. Assume that a rule R: $iset \Rightarrow c$ is used to predict the class label of a transaction that contains $iset$. The accuracy of the prediction is the probability of c being the class label of this transaction:

$$accr(R) = p(c|iset) \tag{5.1}$$

The confidence value of R (i.e., $conf(R)$) is the percentage of transactions that contain class label c among the transactions containing itemset $iset$ in D. Thus, the confidence value of R is the accuracy of the prediction "if $iset$ then c" among the transactions in the training dataset D:

$$conf(R) = p_D(c|iset) \tag{5.2}$$

Statistically, each transaction that contains $iset$ can be considered as a "trial". If a trial belongs to class c, then the outcome is "success", otherwise the outcome is "false". The prediction accuracy of R can then be considered as the probability of success on a random trial, or in other words, the population proportion of all possible trials. On the other hand, the confidence of R represents the success proportion on a trial within the training dataset D which can be considered as a sample set of transactions. Then, the use of confidence for estimating the prediction accuracy can be considered as the use of a sample proportion for estimating the population proportion of a binomial distribution.

Now, we apply proportion estimation [Fre98] which states that population proportion p of a normal distribution can be estimated from sample proportion $\widehat{p}$ (i.e., $conf(R)$) with the standard error of

$$S_E = \sqrt{\frac{\widehat{p}(1-\widehat{p})}{n}} \tag{5.3}$$

Here, n is the sampling size which is the number of transactions containing $iset$ in D. This can be used to explain statistically the use of association rules mined with the support and confidence constraints for classification. When the support value increases, the number of data samples matching the rule increases. This consequently leads to a decrease in the estimation error. Thus, the support threshold value gives the statistical significance of the class association rules [CB91]. It ensures that the accuracy of the rule observed in the training dataset, which is the confidence of the rule, is close to the rule's actual prediction accuracy. The confidence value can then be used for rule selection and classification.

5.1.2 Support and Confidence Thresholds for Rule Discovery

In this section, we discuss the rule mining process using the support-confidence framework which is the most popular and effective approach for mining association rules. In this framework, a support threshold is used to mine the set of frequent itemsets. A confidence threshold is then used to gather the set of association rules from the set of frequent itemsets.

First, let's consider the performance of the rule mining process. The number of frequent itemsets grows exponentially when the support threshold value is decreased. This will lead to an exponential increase in runtime as well as resource usage such as memory or disk space during the frequent itemset mining process [AS94, HP00]. After obtaining the set of frequent itemsets, association rules are generated and filtered using the confidence threshold. This process is simple and straightforward. All the rules that are formed from the resultant frequent itemsets are counted for confidence. The rules with confidence below the confidence threshold value are then discarded. Thus, the confidence threshold does not affect the performance (runtime and resources used) of the mining process. It only affects the number of resultant association rules.

Next, we consider the effects of the support and confidence thresholds on associative classification that is built using the set of resultant rules discovered with the threshold values. The support measure determines the usefulness or "generalization" level of association rules. If the support threshold is set to a large value, many accurate rules with high confidence but small support values may be discarded. On the other hand, if a very small value of the support threshold is set, rules which occur in a few data samples may be obtained. Such rules may not give a good prediction as they occur only in a small number of data samples. Thus, the support threshold value does affect considerably the accuracy of associative classification.

The confidence threshold is used to discard inaccurate rules of small confidence values. Actually, as AC tends to rely on high confidence rules for classifying data samples, the confidence threshold value only slightly affects the accuracy of associative classification. Let's consider the case of associative classification that uses a single rule to classify a data sample. To classify a certain data sample, the "best" rule with the highest confidence value that matches this data sample is selected and used to predict the class of the data sample. If the set of association rules is considered good for AC, the best rule for each data sample should have a high value of confidence. For instance, assume that most of the rules out of the best ones have the confidence value greater than, say, 60%. In this case, association rules with confidence value below 60% are almost certainly not used for classifying any data samples.

Thus, the performance of associative classification will generally be the same whether the confidence threshold value is set as 50% or 60%. In the case of associative classification that is based on multiple rules, the confidence threshold value may have more effects on classification accuracy because all the rules that match a data sample are used to classify it. However, as more accurate rules (i.e., with higher confidence values) play more importance roles (higher weight in classification, for example) in the classification process, the effect of low confidence rules should be small. In other words, the confidence threshold value is not really that important to the performance of AC when there are a sufficient number of rules discovered. In practice, most AC approaches [LHM98, LHP01, LMW01] use the confidence threshold value of 50% with which a sufficient number of rules could be discovered for most of the selected datasets and support threshold values.

In short, the value of the confidence threshold is not really matter to the performance of the rule mining process as well as the accuracy of associative classification when it is set to an appropriate range of values (for example, around 50% for most of the datasets). However, the support threshold value has a strong impact on the performance of the rule mining process as well as the accuracy of associative classification. In the next two subsections we will discuss in more details the effects of the support threshold on the performance of the rule discovery process and the accuracy of the associative classification.

5.1.3 Effects of Support Threshold on Rule Discovery

Let's consider a scenario of mining a set of CARs from a training dataset D. Let $R : iset \Rightarrow c$ be a CAR and D_c be the collection of data samples belonging to class c in D. The support of R is the percentage of data samples that contains $iset$ and belongs to class c with respect to the total number of transactions in D. Thus, given the support $supp_D(R)$ of R in D and the size $|D|$ of D, the number of data samples that contains $iset$ and belongs to class c is $supp_D(R) \times |D|$. This is also the number of data samples that contains $iset$ in D_c which can also be calculated as the product $supp_{D_c}(iset) \times |D_c|$ of the support of $iset$ in D_c and the size of D_c. Thus, $supp_{D_c}(iset) = \frac{supp_D(R) \times |D|}{|D_c|}$. It means that the problem of finding rules in the form of $iset \Rightarrow c$ from D with the support threshold $minSupport$ becomes the problem of finding frequent itemsets $iset$ from D_c with the support threshold of $\frac{minSupport \times |D|}{|D_c|}$. In other words, the problem of discovering the complete set of CARs can be considered as finding frequent itemsets from separate datasets of different classes. The only difference between rule discovery in AC and conventional frequent itemset mining is that the former task may

carry out multiple frequent itemset mining processes for mining rules of different classes simultaneously or counting the support and confidence at the same time [LHM98, LHP01]. Generally, the rule discovery process can be carried out using any association rule mining algorithms and the performance (i.e., runtime and resource usage) depends on the rule mining algorithm it employs.

The experiments given in [AIS93, HP00, HGN00] have shown that, in association rule mining, a decrease in the support threshold leads to an exponential increase on the number of frequent itemsets, which consequently results an exponential increase in runtime and resource usage (i.e., memory and disk space) during the frequent itemset mining process. A similar conclusion was also given by Zheng *et al.* [ZKM01] when they compared the most well-known association rule mining algorithms based on real-world datasets. They have shown that with the support level that generates around one million rules, most of the algorithms are efficient and can complete processing in less than 10 minutes. However, when the support threshold value is set below this level of support, the numbers of frequent itemsets and association rules grow very rapidly, and most of the algorithms will quickly run out of either memory or disk space [ZKM01].

5.1.4 Effects of Support Threshold on Classification Accuracy

In [LHM98, LHP01], it is reported that the support threshold in the range of [1%-2%] gives the highest overall accuracy based on the 26 datasets it has used for performance evaluation. The question is that does the optimal support value in the range of [1%-2%] apply to each dataset of those 26 datasets? Baralis *et al.* [4] pointed out that the optimal accuracy has been achieved with very different support threshold values and the value of support threshold of [1%-2%] may be far from optimal for some datasets. In Section 5.1.1, we have discussed that infrequent rules can also be used in classification for large datasets. In other words, for a large dataset, the support threshold may be set to a low value to obtain accurate but rare rules for classification. These rules will probably help form a more accurate associative classifier. However, a low support value may also lead to a more complex rule discovery process.

From formula (5.3), we can observe that the value of error depends on the sampling size. In turns, the sampling size of a rule depends on the support value and the size of the dataset. For a large dataset, the sample size can be large even with a small support value. This means that, for a large dataset, the support threshold may be set to a low value to obtain

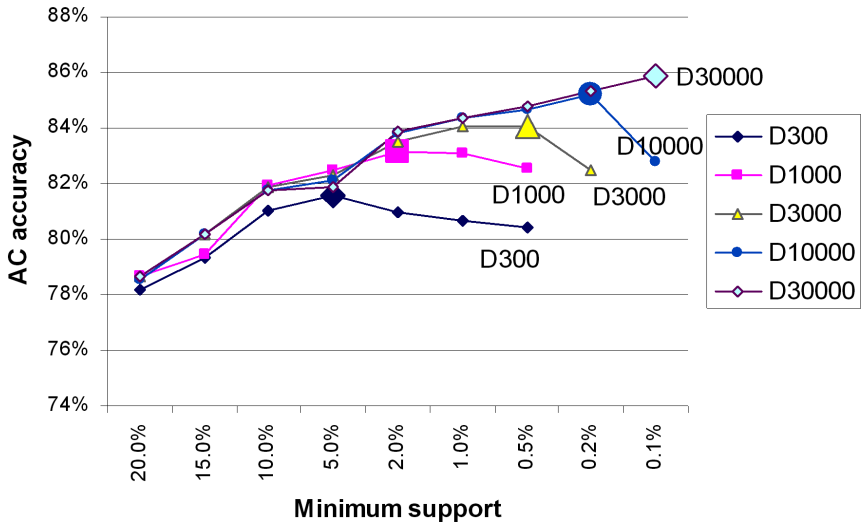


Figure 5.2: Performance of AC with different support threshold values.

accurate but rare rules for classification. These rules will probably help form a more accurate associative classifier. However, a low support value may also lead to a more complex rule discovery process.

We have carried out an experiment to illustrate the relationship between optimal support threshold value and dataset size. In this experiment, associative classifiers are built from different datasets extracted from different subsets of the same dataset. In particular, we extract five datasets, namely D300, D1000, D3000, D10000 and D30000 from the Adult training dataset [NHBM98] that contains 32,561 data samples. Each dataset contains data samples that are randomly extracted from the Adult training dataset. The sizes of the datasets are 300, 1,000, 3,000, 1,0000 and 30,000 respectively.

Figure 5.2 shows the accuracy of the associative classifiers using the five datasets with different values of support thresholds ranging from 0.1% to 20%. From the performance results, we have observed that the associative classifier based on each dataset gathers the optimal (best) accuracy at a certain value of support threshold (which is highlighted in the figure). When the support threshold value moves away from the optimal value (either increasing or decreasing), the classification accuracy decreases accordingly.

Table 5.1 shows the optimal accuracy of the associative classifiers that are built based on the five datasets. The optimal support thresholds and the number of data samples corresponding to the optimal support threshold are given. The results have shown that the

Table 5.1: The optimal performance and support threshold values of AC.

Dataset	Optimal accuracy	Optimal support threshold	Number of data samples corresponding to the optimal support threshold
D300	81.60%	5.0%	15
D1000	83.15%	2.0%	20
D3000	84.09%	0.5%	15
D10000	85.24%	0.2%	20
D30000	85.88%	0.1%	30

associative classifiers built using larger datasets have achieved a higher accuracy. The optimal support value decreases when the dataset size increases. However, the number of data samples corresponding to the optimal support threshold remains quite stable.

The performance results have also shown that very infrequent rules can be used to improve the accuracy of an associative classifier based on a large dataset. For example, the accuracy of the associative classifier based on the last dataset of 30,000 data samples increases when the support threshold is reduced from 0.2% to 0.1%. Thus, rules with support values around 0.1% can help improve the accuracy of this associative classifier.

5.1.5 Discussion

The rule discovery process for AC is computationally similar to association rule mining. Thus, this process is computationally expensive due to the large search space of possible rules. The optimal associative classification accuracy based on different training datasets may have different optimal support values. With the same kind of data, larger training datasets may achieve better accuracy in associative classification in comparison with smaller training datasets. At the same time, the very small optimal support values corresponding to large datasets may lead to a more complex rule discovery process and result with more rules. This also means that there exists a lower bound (called *support threshold lower bound*), below which the support threshold may cause performance degradation of the rule discovery process. That is, there is a tradeoff between classification accuracy and complexity of the rule discovery process in conventional AC approaches for large training datasets. This raises the following issues on associative classification with large training datasets:

- To achieve high accuracy in associative classification, the rule discovery process may be very computational expensive.

- The optimal accuracy may not be achieved if the optimal support threshold is below the support threshold lower bound.

To tackle the problems related to large datasets for AC, in this research we propose two techniques, namely Mstep-AC and AIS-AC, for association rule discovery for AC. We will discuss the Mstep-AC approach in the next section, and the AIS-AC approach in Chapter 6.

5.2 The Mstep-AC Approach

In the rule selection and classification processes, class association rules are ranked according to their prediction accuracy. A rule with a high rank means that it is accurate when the rule is used to classify data samples. On the other hand, a rule with a low rank means it will produce an inaccurate prediction when it is used. If a data sample matches some *high-ranked rules*, the class prediction for the data sample will be accurate. We consider this data sample as a *high-ranked data sample*. In contrast, if a data sample only matches *low-ranked rules*, the class prediction for it will be inaccurate and we call it a *low-ranked data sample*. Thus, in the classification process, most of the errors are occurred due to low-ranked data samples.

To achieve an accurate classification without a complex rule discovery process, we propose the Multiple-step rule discovery for AC (Mstep-AC) approach, which focuses on discovering more and better rules for low-ranked data samples. Let S be the set of discovered rules. We call the rule with the highest rank in S which matches a data sample as the *best rule* of the data sample. We also consider the rank of the best rule of a data sample as the rank of the data sample. That is, given the set of discovered rules S, the rules in S are ranked based on the confidence and support measures as discussed in Section 2.3.2. The set of data samples can then be ranked based on the best rule according to each data sample.

The rule discovery process of the Mstep-AC approach, which is illustrated in Figure 5.3, is described as follows:

1. *Rule Discovery.* Conduct the conventional rule discovery process on the training dataset D. The process returns the set of discovered rules in S. If the number of steps is 1, the process terminates with the resultant ruleset U which is set to S. The multiple-step rule discovery process then becomes the same as the conventional rule discovery process.

2. For each step, repeat 2.1 to 2.4 until the specified number of steps is reached.

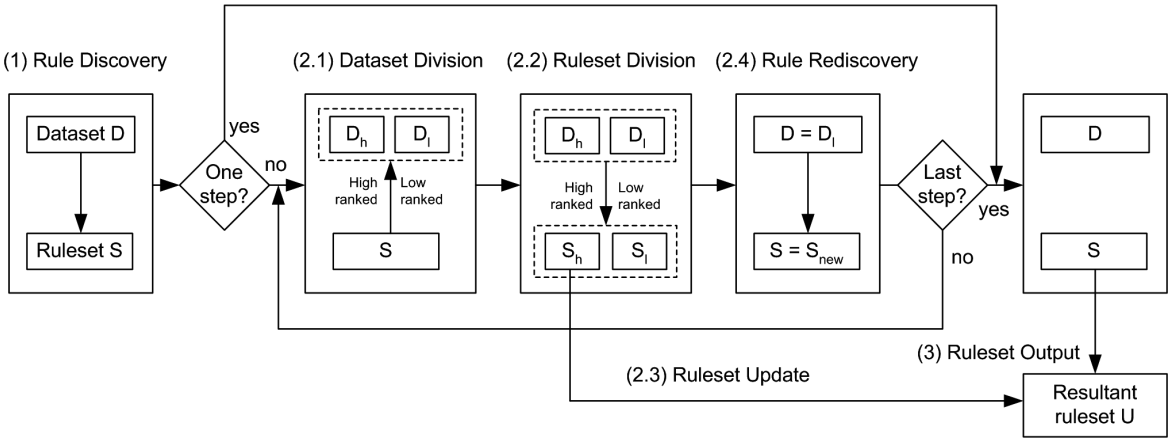


Figure 5.3: The Mstep-AC approach.

2.1. *Dataset Division.* Divide training dataset D into *high-ranked dataset* D_h and *low-ranked dataset* D_l based on ruleset S such that $|D_h| = t \times |D|$ where t is a predefined *dataset reduction rate*. The high-ranked dataset contains high-ranked data samples, whereas low-ranked dataset contains low-ranked data samples.

2.2. *Ruleset Division.* Divide the set of rules in S into *high-ranked ruleset* S_h and *low-ranked ruleset* S_l according to datasets D_h and D_l respectively. The division is carried out such that, if d is the lowest ranked data sample in D_h and R is the lowest ranked rule in S_h, then R will be the best rule for d.

2.3. *Ruleset Update.* Insert high-ranked ruleset S_h into the resultant ruleset U.

2.4. *Rule Rediscovery.* Remove high-ranked dataset D_h from D. Carry out the rule discovery process from the updated training dataset D. This step then produces another set of discovered rules S_{new} that is set to S.

3. *Ruleset Output.* Insert the last ruleset S into U and return the resultant ruleset U.

Figure 5.4 illustrates the division of the training dataset D into high-ranked dataset D_h and low-ranked dataset D_l, and the division of the set of discovered rules S into high-ranked ruleset S_h and low-ranked ruleset S_l. The data sample d and the best rule R of it are at the boundaries of the two datasets and two rulesets respectively. Because d is the lowest ranked data sample in D_h, each data sample in D_h must match at least one rule in S_h which is ranked higher than R. As the rank of rules in S_h are higher than that in S_l, data samples in D_h will probably be classified by rules in S_h even though they may match rules in S_l. In contrast, as the rank of data samples in D_l is lower than that of d, data samples in D_l will

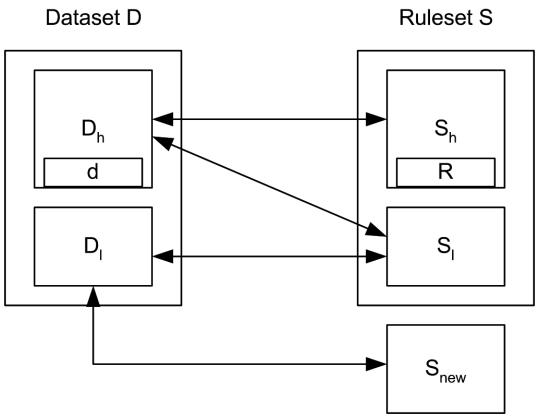


Figure 5.4: Dataset and ruleset division.

not match any rules in S_h. In other words, data samples in D_l will be classified only by rules in S_l.

Our intention is to use the newly discovered ruleset S_{new} for classifying low-ranked data samples in D_l. Thus, S_l is replaced by S_{new}. Let's compare performance of the sets of rules S_l and S_{new} for the classification process on the data samples in D_l. First, we consider the confidence measure. The confidence of the rules in S_l is counted on the training dataset D, while the confidence of rules in S_{new} is counted on D_l. As the sets of rules (i.e., S_l and S_{new}) are only used for classifying data samples in D_l, the confidence of rules in S_{new} is more accurate in presenting the rules' prediction accuracy.

Next, let's consider the support measure. Here, our heuristic is that if a rule R is more appropriate for classifying data samples in D_l than D_h, then it will occur more frequently in D_l than D_h. Let's denote $supp(R)$, $supp_{D_h}(R)$ and $supp_{D_l}(R)$ as the support values of rule R counted on datasets D, D_h and D_l respectively. That is, appropriate rules for classifying data samples in D_l will satisfy the following condition:

$$supp_{D_h}(R) < supp_{D_l}(R) \tag{5.4}$$

The number of data samples in D that supports rule R is equal to the total number of data samples supporting R in both D_h and D_l. Thus,

$$supp(R) \times |D| = supp_{D_h}(R) \times |D_h| + supp_{D_l}(R) \times |D_l| \tag{5.5}$$

$$supp(R) = supp_{D_h}(R) \times t + supp_{D_l}(R) \times (1-t) \tag{5.6}$$

$$t = \frac{supp(R) - supp_{D_l}(R)}{supp_{D_h}(R) - supp_{D_l}(R)} \tag{5.7}$$

where t is the predefined dataset reduction rate.

From formula (5.4), if a rule R satisfies condition (5.1), then we will have $supp(R) < supp_{D_l}(R)$ as t is a positive number. In other words, the support value of R in D_l is larger than that in D. This can imply that if R has been discovered in the first step using dataset D (i.e., $minSupport < supp(R)$), it will also be discovered in the next step using dataset D_l (i.e., $minSupport < supp_{D_l}(R)$). Moreover, even if R is not discovered in the first step, it may still be discovered in the second step (i.e., $supp(R) < minSupport < supp_{D_l}(R)$).

Thus, the set of new discovered rules S_{new} contains more rules suitable for classifying data samples in D_l in comparison with S_l. We may say that the set of discovered rules less underfits the data as the rules are generated and assessed in the appropriate range of data samples in which the rules are probably used for classifying. This means that Mstep-AC approach would construct are more accurate associative classifier compared with conventional AC approaches. Let's consider the complexity of the rule discovery process with Mstep-AC. As the different steps use the same support and confidence threshold values, the search space of rules is similar. The runtime of any subsequent steps should be less than the previous one as the dataset size is getting smaller after each step. That is, with n steps, the runtime would be less than n times of that of CBA.

Next, we discuss the effect of various parameters such as the number of steps s and the reduction rate t to the performance of the Mstep-AC approach. Consider the two steps (i.e., $s = 2$) of rule division in Figure 5.4 that divides the dataset into two subsets D_h and D_l. The corresponding sets of rules are then generated from the two subsets of data. When the reduction rate t gets near to 0% or 100%, one of the subsets (i.e., D_h when $t \approx 0\%$ or D_l when $t \approx 100\%$) will get smaller. This would lead to an overfitting learning from the small subset. On the other hand, when the subset is too small, the rules generated from it may affect slightly to the overall performance as the applicability of the rules is limited to only a small number of data samples. Thus, we may expect that the performance of Mstep-AC2 is good when t is between the range from 0% to 100%, and the performance decreases and closes to that of CBA when t is getting near to one of these two ends.

When the number of steps increases, the divided subsets get smaller. According to the mechanism of rule generation and confidence calculation presented above, we may expect that the accuracy of Mstep-AC increases due to the decrease on overfitting data. However, when the number of steps continues to get larger, the set of discovered rules, which consists of many subsets of rules, each of which generated and assessed from a small partition of the

dataset, may start overfit the data. Thus, we may expect that the performance would go down at a certain number of steps. As such, the reduction rate determines the range of data in the dataset that the learning process focuses on. The learning process would focus on high-ranked data when the reduction rate is small. On the other hand, the learning process would focus on low-ranked data when the reduction rate is large.

The implementation of the Mstep-AC algorithm is shown in Algorithm 5.1. The inputs include the training dataset D, the support and confidence thresholds *minSupport* and *minConfidence*, the dataset reduction rate t and the number of steps s. The output is the set of discovered rules U. *Rule Discovery* is carried out in the first step using the conventional rule discovery process (in line 3) which returns the set of discovered rules in S.

For each subsequent step of the Mstep-AC approach, *Dataset Division* and *Ruleset Division* are carried out by specifying the separators d_t and R_t that divide the training dataset D and the set of discovered rules S into the high-ranked and low-ranked datasets and rulesets respectively. To do this, the best rule in S corresponding to each data sample in D is identified and the rank of each data sample in D is set to that of its best rule (in lines 6-10). When D is sorted (in line 11) in decreasing order according to the rank of data samples, the first $(t \times |D|)$ rules of it are obviously the set of high-ranked data samples D_h. Thus, d_t is assigned to be the $(t \times |D|)^{th}$ rule in the sorted data samples in D (in line 12). The separator rule R_t of S, which is the best rule of d_t, is also determined (in line 13). *Ruleset Update* then inserts the rules of S_h, which are also the rules in S with ranks greater than or equal to that of R_t, into the resultant set U (in lines 15-19). In *Rule Rediscovery*, it eliminates the data samples in D_h (in line 21), which is also the set of first $(t \times |D|)$ rules in the sorted data samples of D, from D. The dataset reduction rate is, therefore, equal to $\frac{t \times |D|}{|D|} = t$.

The conventional discovery process is then carried out again on the new training dataset D with the same support and confidence threshold values *minSupport* and *minConfidence* until the number of steps is reached. Finally, the *Ruleset Output*, which inserts the last ruleset S into U and returns the resultant ruleset U, is carried out (in lines 25-26).

5.3 Performance Evaluation

The performance of the proposed Mstep-AC approach is evaluated using a 3.0 GHz Pentium PC with 1 GB of memory running with the Microsoft Windows XP. We have implemented the Mstep-AC algorithm, and the CBA and msCBA algorithms [LHM98, LHP01, LMW01] for comparison. In the conventional rule discovery process, most of the AC approaches follow

Algorithm 5.1 Mstep-AC for Associative Classification.

Input:

D - training dataset
$minSupport$ - minimum support threshold value
$minConfidence$ - minimum confidence threshold value
t - dataset reduction rate
s - number of steps

Output:

U - set of final discovered rules

Process:

1: Initialize $U \leftarrow \emptyset$
2: // 1 - Rule Discovery
3: $S = ruleDiscovery(D, minSupport, minConfidence)$
4: **for** $step = 2$ to s **do**
5: // 2.1, 2.2 - Dataset and Ruleset Division
6: **for all** data sample d in D **do**
7: $S_m = \{R \mid (R \in S) \wedge (R \ matches \ d)\}$
8: Set R_b to the rule with highest rank in S_m
9: $d.rank = R_b.rank$
10: **end for**
11: Sort D in decreasing order of the rank of data samples
12: Set d_t to the $(t \times |D|)^{th}$ data sample of D
13: Set R_t to the best rule of d_t
14: // 2.3 - Ruleset Update
15: **for all** rule R in S **do**
16: **if** $R.rank \geq R_t.rank$ **then**
17: Insert R into U
18: **end if**
19: **end for**
20: // 2.4 - Rule Rediscovery
21: Remove first $(t \times |D|)$ data samples from D
22: $S = ruleDiscovery(D, minSupport, minConfidence)$
23: **end for**
24: // 3 - Ruleset Output
25: Insert S into U
26: **return** U

the simple scheme of the CBA algorithm proposed by Liu *et al.* [LHM98] in which class association rules are mined based on a single support threshold value. To compare Mstep-AC with CBA, we also use a single support threshold in Mstep-AC. In the implementation of the Mstep-AC algorithm for the first two experiments, the dataset reduction rate was set to 0.5 and the number of steps was set to 2 (denoted as Mstep-AC2) and 3 (denoted as Mstep-AC3).

5.3.1 Datasets

We have used the datasets obtained from the UCI Machine Learning Repository [NHBM98] for evaluating the performance of the proposed Mstep-AC approach. There are about 70 datasets in the repository. We have selected large datasets from the repository for performance evaluation. Thus, we have chosen four attribute datasets, namely *Adult*, *Digit*, *Letter* and *Nursery*, with each consisting of more than 10,000 data samples. Apart from measuring the accuracy of associative classification, this also aims to measure the scalability of the algorithms when mining the association rules. In addition, the use of larger dataset size in performance measurement will also make the evaluation more credible and reliable. For example, if a dataset contains only a few hundreds of samples (as in the case of many datasets in the repository), classification results of several data samples in the dataset may influence significantly the performance of the classifier, which might not be desirable. Further, the support value of 1% in the datasets of small sizes may include rules that consist of only several data samples, leading to unreliable results during rule selection. The discretization process, which discretizes continuous data into intervals and then maps them into items, is performed using the Entropy method from the MLC++ machine learning library [KJL+94].

Table 5.2 lists the properties of the four attribute datasets. The *Adult* and *Digit* datasets have already been separated into training and test sets in the repository. For the *Letter* and *Nursery* datasets, they are not stored as separate training and test sets. Therefore, we divide the data from the datasets randomly into training and test sets, in which the training set is about twice the size of the test set.

In addition, we have also used a textual dataset, called *Reuters-R8* [CC], for evaluating the performance of the proposed Mstep-AC approach. The Reuters-R8 dataset has been described in Section 4.2.4 of Chapter 4.

Table 5.2: Attribute Datasets.

Dataset	No. of attributes	No. of classes	No. of samples	Training set	Test set
Adult	14	2	48,842	32,561	16,281
Digit	12	10	10,992	7,494	3,498
Letter	16	26	20,000	13,333	6,667
Nursery	8	5	12,960	8,640	4,320

5.3.2 Experimental Results for Attribute Datasets

We have carried out the first experiment for evaluating the performance of the AC approaches using the textual dataset Reuters-R8. Figure 5.5 shows the performance results of the Mstep-AC and CBA approaches in terms of accuracy with minimal support thresholds ranging from 0.2% to 1% for all algorithms. We can see that the Mstep-AC approach has achieved better accuracy in comparison with the conventional CBA approach for all four datasets with all support thresholds. In addition, Mstep-AC3 has achieved better performance in comparison with Mstep-AC2. Table 5.3 shows the runtime and maximal number of candidate rules generated during the rule discovery process for the CBA and Mstep-AC approaches. The number of candidate rules can be used to estimate the amount of memory needed for each approach. We can see that the amount of main memory used in Mstep-AC is quite close to that of CBA. This means that the amount of main memory used in subsequent steps of Mstep-AC is generally not more than that in the first step (i.e., CBA). In addition, as discussed in Section 5.2, the subsequent steps of Mstep-AC are faster than their predecessors. Thus, the runtime of Mstep-AC is not more than s times of that of CBA (i.e., the first step) with s as the number of steps. Table 5.3 shows that the runtime of Mstep-AC2 and Mstep-AC3 with the same support threshold value is about 1.5 and 2 times of the runtime of CBA respectively.

Generally, the Mstep-AC approach has performed better than CBA. For example, the performance results from the *Adult* dataset have shown that Mstep-AC2 with support threshold of 0.8% has achieved the same level of accuracy of around 85.7% as CBA with support threshold of 0.2%. However, Mstep-AC2 runs faster (16.3 minutes vs. 50.3 minutes) and uses less memory (36,762 candidate rules vs. 182,307 rules) than CBA with these threshold values. Moreover, let's consider the three datasets *Adult*, *Digit* and *Nursery*. In these datasets, CBA has achieved the best accuracy with the support threshold of 0.2%. However, Mstep-AC3 can achieve even better accuracy with the support threshold of 1% and 0.8%, and, at the

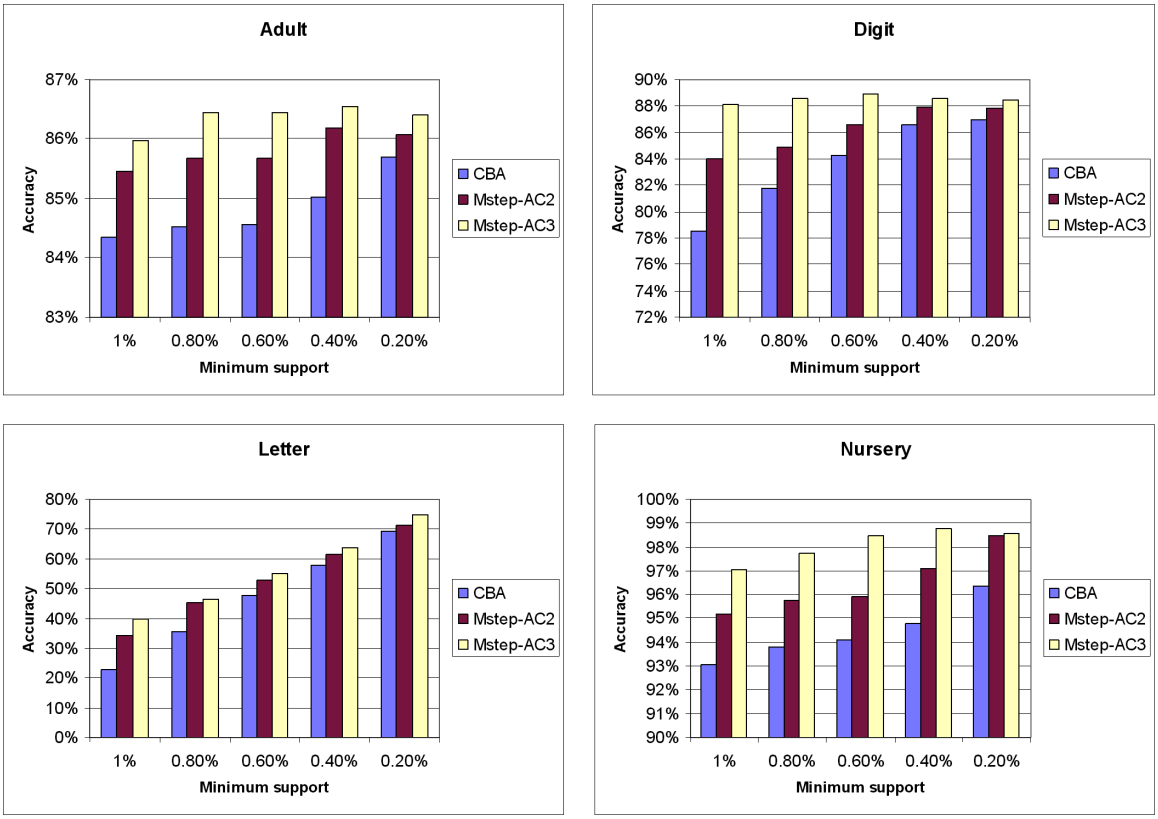


Figure 5.5: Performance comparison of Mstep-AC and CBA.

same time, it is faster and uses much less memory in comparison with CBA with the support threshold of 0.2%.

In addition, we also compare Mstep-AC with msCBA, a multiple class support thresholds version of CBA. For a fair comparison, Mstep-AC has also been implemented using multiple class supports. We set the total minimal support threshold (i.e., t_minSupport) ranging from 1% to 5%. Note that, the total minimal support threshold is equal to the sum of the minimal support thresholds of all different classes in the dataset. Thus, for example, if the t_minSupport for the dataset *Digit*, which contains 10 classes, is 2%, then the average minimal support threshold of CARs of each class is about 0.2%, which is quite small. Note that, the dataset size at the third step of Mstep-AC3 is about one quarter of that of the original dataset with the dataset reduction rate of 0.5. And the t_minSupport of 1% is equivalent to about 1 to 3 transactions at the third step of Mstep-AC3 for the Digit and Letter datasets (which contain 10 and 26 classes respectively). This number of transactions is too small to warrant reliable prediction. Therefore, with the t_minSupport of 1%, the performance of Mstep-AC3

Table 5.3: Runtime (in minutes) and memory usage of Mstep-AC and CBA.

Dataset	Approach	1%		0.8%		0.6%		0.4%		0.2%	
		Run time	#of rules	Run time	#of rules	Run time	#of rules	Run time	#of rules	Run time	#of rules
Adult	CBA	13.8	26575	16.3	36762	20.1	53616	26.8	87030	50.3	182307
	Mstep-AC2	24.1	26575	28.3	36762	34.7	53616	46.2	87030	85.5	182307
	Mstep-AC3	32.2	26575	37.8	36762	46.7	53616	62.6	87030	113.6	182307
Digit	CBA	0.7	1737	0.9	2988	1.4	5484	3.0	12517	14.0	43938
	Mstep-AC2	1.0	2317	1.3	4444	2.1	8046	4.9	18170	23.9	45970
	Mstep-AC3	1.1	2317	1.6	4444	2.7	8046	6.7	18170	43.4	55410
Letter	CBA	0.8	105	1.0	362	1.5	1835	2.1	2362	4.8	7247
	Mstep-AC2	1.1	105	1.5	362	2.0	1835	2.8	2362	6.3	8413
	Mstep-AC3	1.3	105	1.7	362	2.2	1835	3.1	2982	7.2	8413
Nursery	CBA	0.2	1606	0.3	2694	0.3	3962	0.4	5521	0.5	8488
	Mstep-AC2	0.4	2874	0.4	3747	0.5	4557	0.6	5521	0.9	9388
	Mstep-AC3	0.4	2874	0.5	3747	0.6	4557	0.8	5521	1.2	9388

is obtained without the third step (i.e., equivalent to Mstep-AC2) for the Digit and Letter datasets.

As shown in Figure 5.6, the Mstep-AC approach has achieved better accuracy in comparison with the msCBA approach. The classification accuracy has been improved considerably for all four datasets with all support thresholds using the Mstep-AC approach. In addition, Mstep-AC3 has achieved better performance in comparison with Mstep-AC2.

The performance results have also shown that the optimal support threshold values of CBA and msCBA approaches for the four datasets are all very small. For example, the classification accuracy gradually increases when the support threshold value reduces from 1% to 0.2%, as shown in Figure 5.5. It has suggested that the optimal support threshold values of the CBA approach for all four datasets are below 0.2%. Similarly, Figure 5.6 has shown that the optimal values of the total minimum support threshold of the msCBA approach on the three datasets, namely *Adult, Letter* and *Nursery,* are less than 1%. These small optimal values of support threshold may not be feasible with limited runtime and system resources. In contrast, Mstep-AC3 can achieve optimal accuracy within the selected ranges of the support threshold for the *Adult*, *Digit* and *Nursery* datasets with single class support and for the *Digit, Letter* and *Nursery* datasets with multiple class supports. In other words, the optimal value of support threshold of the Mstep-AC approach is probably larger than that of the CBA approach. This enables Mstep-AC to achieve better classification accuracy within limited runtime and system resources.

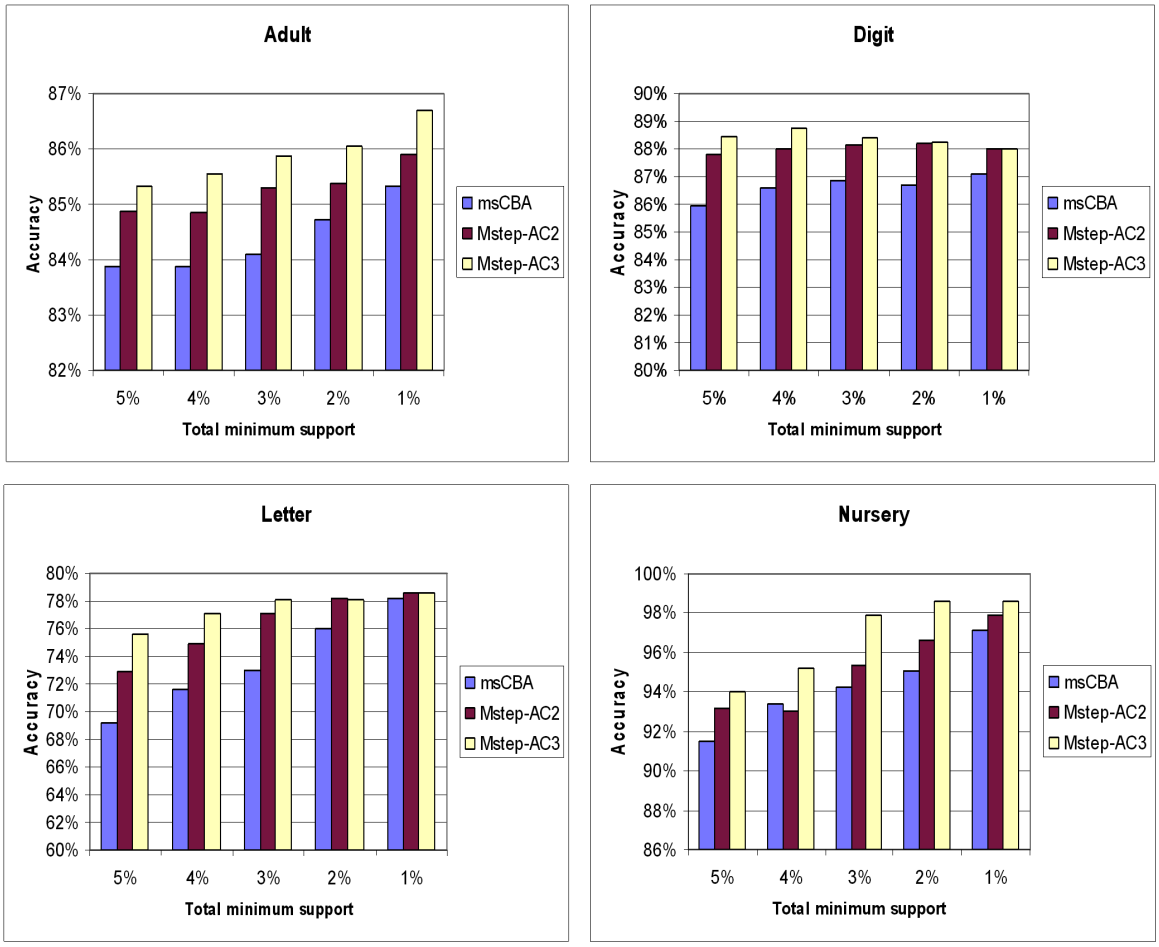


Figure 5.6: Performance comparison of Mstep-AC and msCBA.

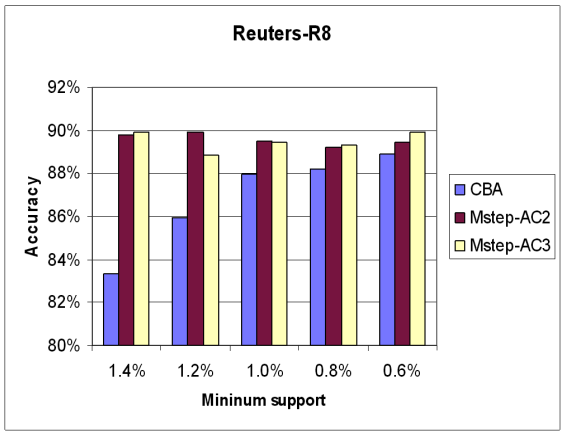
Table 5.4 shows the performance improvements of Mstep-AC over CBA and msCBA in terms of average accuracy of different support threshold values. The improvements range from 0.9% to 6.4% for Mstep-AC2 and from 1.5% to 9.3% for Mstep-AC3.

5.3.3 Experimental Results for Textual Dataset

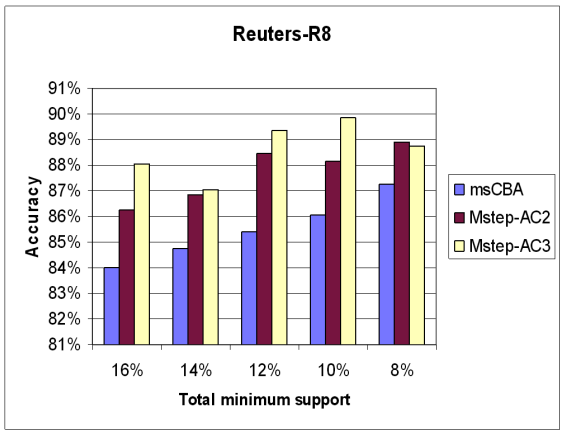
We have carried out the second experiment for evaluating the performance of the AC approaches using the textual dataset Reuters-R8. Figure 5.7(a) shows the performance results of the Mstep-AC and CBA approaches which use single class support and Figure 5.7(b) shows the performance results of the Mstep-AC and msCBA approaches which use multiple class supports. Using the textual dataset, a large number of frequent itemsets and rules can be generated even with moderate support threshold values. For example, over one million frequent itemsets are generated for some classes when the support threshold value is set to 0.4% with the CBA algorithm and the total support threshold value is set to 6% for the msCBA

Table 5.4: Average performance improvements.

Approach	Compared with	Performance improvement			
		Adult	Digit	Letter	Nursery
Mstep-AC2	CBA	1.0%	2.6%	6.4%	2.1%
Mstep-AC3	CBA	1.5%	4.9%	9.3%	3.7%
Mstep-AC2	msCBA	0.9%	1.4%	2.8%	0.9%
Mstep-AC3	msCBA	1.5%	1.7%	3.9%	2.6%



(a) with single class support



(b) with multiple class supports

Figure 5.7: Performance comparison of Mstep-AC with CBA and msCBA using the textual dataset.

algorithm. With these support values, it is likely that the implemented algorithms will take a very long runtime and probably running out of main memory. Thus, in this experiment, we consider the minimum support threshold from 0.6% to 1.4% for rule discovery with single class support. For rule discovery with multiple class supports, the total minimum support threshold is considered from 8% to 16%.

Figure 5.7 shows similar results to that of comparing Mstep-AC with the CBA and msCBA algorithms in the attribute datasets. Generally, Mstep-AC has achieved considerably higher accuracy in comparison with CBA and msCBA for all selected values of the support threshold. In addition, Mstep-AC can achieve good accuracy with considerably large support threshold values in comparison with CBA and msCBA. In particular, Mstep-AC has achieved the best accuracy with the minimum support value of 1.2% and 1.4% for single class support and the total minimum support of 10% and 12% for multiple class supports. It seems that the optimal support threshold values of the CBA and msCBA approaches are below the selected range of

Table 5.5: Performance of Mstep-AC2 in comparison with CBA with different reduction rates.

Dataset	CBA	Reduction rate									
		10%	20%	30%	40%	50%	60%	70%	80%	90%	95%
Adult	84.6%	84.7%	85.1%	85.0%	85.2%	85.7%	86.3%	85.9%	85.4%	85.0%	84.6%
Digit	84.3%	84.3%	86.2%	86.5%	86.9%	86.6%	86.8%	87.7%	87.3%	87.2%	86.4%
Letter	47.6%	51.2%	52.2%	53.3%	54.2%	53.8%	54.6%	52.9%	53.2%	53.0%	53.2%
Nursery	94.1%	94.1%	94.3%	95.4%	95.8%	95.9%	96.6%	98.0%	99.2%	97.5%	95.5%
Average increase	0.0%	1.0%	1.8%	2.4%	2.9%	2.9%	3.5%	3.5%	3.6%	3.1%	2.3%

the support threshold. Thus, the Mstep-AC approach can achieve optimal performance with higher optimal support threshold in comparison with the CBA and msCBA approaches. It implies that the rule discovery process of Mstep-AC is more efficient in terms of runtime and resource usage.

5.3.4 Experimental Results on the Effects of Parameter Values

The third experiment has been carried out for evaluating the effects on the values of the parameters of Mstep-AC to its performance. In this experiment, we have used the four attribute datasets mentioned earlier. Mstep-AC was implemented using single class support with support value set to 0.6%.

Firstly, we run Mstep-AC using two steps (i.e., denoted as Mstep-AC2) with different reduction rates. The performance results are shown in Table 5.5. We observe that Mstep-AC2 performs better in comparison with CBA when the reduction rate is between 0% and 100%. This has been analyzed in Section 5.2. In fact, the optimal performance of Mstep-AC2 is achieved with a reduction rate in the range from 60% to 90%. This can be explained as follows. Most of the errors are occurred in the classification process with very low-ranked data samples. That is, even with a high value of reduction rate (say 70%), a large number of errors still occurs in low-ranked data samples (i.e., 30% of data). That is, a more appropriate rule generation and assessment from this low-ranked subset of data still helps increase the performance of the associative classifier considerably. The overfitting problem only affects the result with reduction rate above 80% where the accuracy has started decreasing.

Next, we run Mstep-AC with different numbers of steps. The corresponding reduction rate is set so that the set of data samples retained for the last step is not too small to affect the performance of the classifier. In particular, we set the parameters so that in the last step, the size of the remaining dataset is equal to 20% of that of the original dataset, which is the

Table 5.6: Performance of Mstep-AC in comparison with CBA with different numbers of steps.

Dataset	CBA	Number of steps									
		2	3	4	5	6	7	8	9	10	15
Adult	84.6%	85.4%	86.5%	86.3%	86.3%	86.4%	85.9%	86.3%	86.5%	86.5%	86.4%
Digit	84.3%	87.3%	88.7%	88.9%	88.7%	89.2%	89.8%	89.1%	89.4%	89.2%	89.2%
Letter	47.6%	53.2%	56.1%	56.0%	57.6%	58.4%	59.4%	59.4%	59.7%	61.4%	58.7%
Nursery	94.1%	99.1%	98.6%	98.8%	98.6%	99.0%	98.5%	98.8%	98.8%	98.8%	98.9%
Average increase	0.0%	3.6%	4.9%	4.9%	5.2%	5.6%	5.8%	5.8%	6.0%	6.4%	5.7%

rate that Mstep-AC2 has achieved the best performance. Let s be the number of steps and r be the reduction rate. The number of times that the dataset is reduced is $s-1$. The size of the remaining dataset would be $(1-r)^{s-1}$ after $(s-1)$ times in each of which the dataset is reduced by $(1-r)$. Thus, $(1-r)^{s-1} = 0.2$. Or the reduction rate is chosen such that $r = 1 - 0.2^{1-s}$.

The performance of Mstep-AC with different numbers of steps is shown in Table 5.6. As shown in the table, the average performance increases for the small numbers of steps (i.e. less than 10 in this case). This has been explained that as the number of steps increases, the set of discovered rules is less underfitting the data which probably help improve the performance of Mstep-AC. However, when the number of steps increases to a certain value (i.e., 15), too many data divisions may lead to a very little difference between the datasets, which may eventually cause overfitting in the learning process, thereby probably causing the performance to decrease.

5.4 Summary

Large training datasets may lead to a more accurate associative classification in comparison with small training datasets. However, the rule discovery process for AC is computationally expensive due to the large search space of possible rules. The very small optimal support values corresponding to large datasets may lead to a much more complex rule discovery process and a huge set of discovered rules. In this chapter, we have proposed the Mstep-AC approach for rule discovery for AC. The experimental results have shown that the proposed approach has achieved better classification accuracy while maintaining the rule discovery process to the same level of complexity comparable to conventional AC approaches.

Although Mstep-AC has achieved a considerable improvement in comparison with conventional AC approaches in dealing with large datasets, it may still face the problem of increasing complexity when the size of the training dataset increases. When the dataset size increases, the optimal support threshold value of conventional AC approaches may probably decrease. The optimal support threshold value of Mstep-AC may also decrease (though not to the extent of conventional AC approaches). This means that there is an increase in the complexity of the rule discovery process using either Mstep-AC or conventional AC approaches. In the next chapter, we will propose an approach which can effectively discover the class association rules with very small support threshold values.

Chapter 6

Artificial Immune System for Rule Discovery in Associative Classification

The Artificial Immune System (AIS) [dCVZ99, dCT03, DZ03] is inspired from natural immune systems. The powerful information processing capabilities of the immune system such as feature extraction, pattern recognition, learning, memory, and its distributive nature provide rich metaphors for its artificial counterpart. Specifically, three immunological principles are primarily used in AIS methods [DZ03]. These include the immune network theory, negative selection mechanism, and clonal selection principle. In this chapter, we focus only on investigating the clonal selection principle for data classification. The clonal selection algorithm [dCVZ00a, dCVZ02] follows the population-based search model of evolutionary algorithms that have the capability of dealing with complex search spaces [dCVZ00a, dCT03].

In this chapter, we propose an AIS-based supervised leaning approach, which is called AIS-AC, for discovering association rules for associative classification. The purpose of the proposed approach is to reduce the computational complexity of the rule discovery process while maintaining the accuracy comparable to that of conventional AC approaches. Instead of searching for association rules using conventional association rule mining algorithms and then selecting a suitable subset for the classification process, AIS-AC searches for a subset of association rules for the classification process in an evolutionary manner. Thus, the proposed approach avoids exhaustive searching in the rule discovery process. This is especially important for mining association rules from large datasets with small support threshold values, in which the search space of rules is huge. AIS-AC searches for association rules using the AIS

principle and a set of constraints which prioritize accurate rules for classification. During the search process, the constraints help restrict the regions of accurate rules, whereas the AIS principle helps explore optima of these regions. Thus, we may expect that, if the constraints and the AIS principle are implemented sufficiently for their purposes, the set of rules will be appropriate for AC.

The rest of this chapter is organized as follows. We first review the related work on artificial immune systems. Then, our proposed AIS-AC approach for associative classification and its performance are given. Finally, we give a summary of the chapter.

6.1 Artificial Immune System

6.1.1 Immune System

An immune system [N.K73] is seen as a complex of cells, molecules and organs that protect organisms against infections. One of the fundamental roles of such a system is to recognize and eliminate disease-causing agents known as *pathogens*. An immune system recognizes pathogens indirectly via molecules called *antigens*, a small portion of the pathogen. *B-cells*, which are a kind of immune cells, depend on its protein molecules called *antibodies* to recognize antigens. A B-cell recognizes the antigen when its antibodies come into contact with an antigen of complementary shape [dCVZ99]. The B-cell is also said to stimulate the antigen.

For a given antigen, there are some B-cells that are sufficiently stimulated through high affinity. These B-cells then rapidly produce *clones* with *mutations* at particular sites in the gene. The mutations help produce higher affinity cells, and the proliferation matches better for the antigen with successive generations. Some B-cells of a high affinity to the antigen mature into *memory* cells, which retain the immunological memory of the stimulating antigen. This gives the immune system the capability of producing high affinity antibodies, pre-selected for the specific antigen that has stimulated the primary response [dCVZ02].

6.1.2 Clonal Selection Algorithm

Artificial immune systems are developed based on metaphors from the immune system. The current artificial immune systems observe and adopt immune functions, models and mechanisms, and apply them to solve the problems in computing, engineering and other research areas [dCVZ99]. The three immunological principles including the immune network theory,

negative selection mechanism and clonal selection principle are most frequently adopted in AIS. For example, the Artificial Immune Network (AIN) models [TNH00, TN01], which are based on the immune network theory, simulate a network of interconnected B-cells for antigen recognition. With the capability of data normalization and stimulation, and resource allocation [DZ03], AIN is a powerful approach for pattern recognition and knowledge discovery. Forrest *et al.* proposed the Negative Selection Algorithm (NSA) [FPAC94] which inherits from the negative selection mechanism the ability to detect unknown antigens [DZ03]. This algorithm can be used effectively for change or anomaly detection [DF99].

In [dCVZ00a, dCVZ02], de Castro and Von Zuben proposed an algorithm called CLONALG that focuses on the clonal selection principle and affinity maturation process of adaptive immune response. Figure 6.1 shows the schematic representation of the clonal selection model which can be described in the following steps:

1. The algorithm starts with an initial set of population P_r and an empty memory set M.
2. The *selection* process then selects n best cells to generate a new population P_n according to the affinity principle.
3. The *clonal* process reproduces a population of clones C from the population of P_n cells. This step produces more offspring for higher affinity cells.
4. The *maturation* process mutates the cells to create the population C^*.
5. The *reselection* process reselects the improved cells from C^* and updates the memory set M.
6. The *diversity introduction* process replaces d cells with new ones N_d.

During the maturation process (in step 4), it assigns a lower mutation rate for higher affinity cells than lower affinity cells. The idea is that the cells close to a local optimum need only fine-tuning, whereas cells far from that optimum should move in larger steps towards it or other regions of the affinity landscape [dCT03]. In step 6, the lower affinity cells will have higher probability of being replaced. The selection process leads the population towards more stimulated to antigens, while the maturation and diversity introduction processes help maintain the diversity of the population.

The CLONALG algorithm is similar to mutation-based evolutionary algorithms and has good features for optimization and searching. These features include (i) exploitation and exploration of the search space; and (ii) capability of allocating multiple optima and maintaining

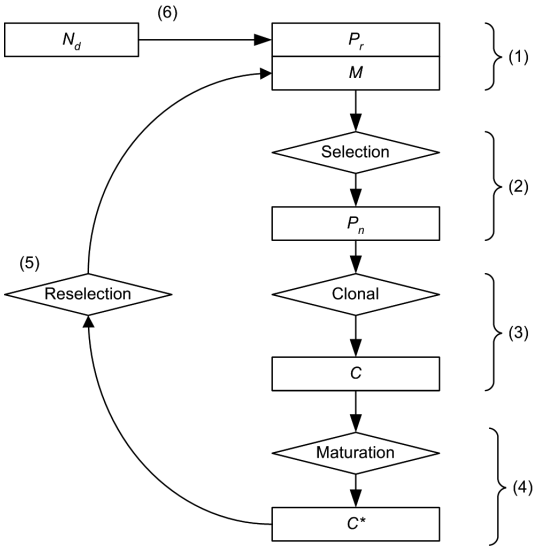


Figure 6.1: The schematic representation of the CLONALG algorithm.

local optimal solutions [DZ03]. Although CLONALG is characterized as an evolutionary-like algorithm, there are some important differences between CLONALG and the typical evolutionary algorithm, Genetic Algorithm (GA) [Hol92]. Inspired by the immune system, CLONALG performs not only proportionate selection, but also affinity inversely proportional hypermutation, and there is no crossover operation [dCT03]. In addition, in [dCVZ00a], de Castro and Von Zuben have shown that CLONALG can reach a diverse number of local optimal solutions, while GA tends to polarize the whole population of individuals towards the best candidate solution.

6.1.3 Artificial Immune System for Data Mining

AIS algorithms have been used in a variety of applications such as computer security [Das99], adaptive control [KN99], fault detection [BT00], etc. Recently, research work has been carried out to employ AIS algorithms for data mining applications such as classification, clustering and frequent itemset discovery.

In [WB02a], Watkins and Bogess proposed a classification model, called AIRS (Artificial Immune Recognition System), which is based on artificial immune network models [TNH00, dCVZ00b]. The AIRS approach employs the immunology metaphors such as resource competition and clonal selection for data classification [dCT02]. Given a dataset, the classifier produces a set of limited resources that can generalize the dataset. The classification can then be carried out based on the set of limited resources, which may be significantly

smaller than the original dataset [WB02b]. In another work, Potter and De Jong [PDJ88] proposed an immune-based approach for the binary classification problem. In the proposed approach, antigens are represented as simple binary strings, and antibodies are represented as binary schemes. A linear matching function, which returns the percentage of matching bits between antigens and antibodies, is used to compute their affinity [dCT02]. The antibodies, with binary representations, can match a wide range of antigens. Thus, an antibody or its B-cell may be present for a group of antigens or data samples, thereby enabling data generalization. The approach can also evolve with a genetic algorithm to maintain diversity of cells [PDJ88].

In [TN01], Timmis and Neal proposed an artificial immune network for unsupervised learning. The proposed network consists of a set of B-cells. Each B-cell, which is connected to other B-cells, is selected according to its affinity with other cells received from the environment. A resource limited mechanism is then applied to control the population of B-cells of the network [dCT02]. The experimental results have shown that the network model is successful in clustering the selected dataset. In [dCVZ00b], de Castro and Von Zuben proposed an artificial immune network model, called aiNet, for data clustering. In aiNet, the clonal selection principle is employed to control the amount and shape of the network cells while the immune network theory is employed to control the network structure [dCVZ00b]. The proposed network model is capable of filtering out redundant data, and describing the data structure and its cluster inter-relationships [dCVZ00b]. In [NGC+03], Nasraoui *et al.* introduced a Dynamic Weighted B-cell (D-W-B-cell) in which an additional time dimension is added into the classical B-cell of AIN models. The ability to remove outdated patterns helps reduce storage and computational costs [DZ03]. The proposed system has been shown to be scalable and efficient for clustering complex real-world data such as Web log data [NCRG03, NCR04].

In [NK02], Nasraoui *et al.* proposed an AIS model for frequent itemset discovery based on the immune network theory. The proposed approach does not use the conventional support measure to determine frequent itemsets. Instead, the support measure is specified based on a "stimulation level" which is based on a relative overlapping level between the itemset and the training data samples. The experiments conducted using a set of Web log data have shown that the AIS-based approach is sufficient to deal with large variability and sparsity of data in comparison with conventional AC approaches for frequent itemset discovery [NK02].

6.2 AIS-based Rule Discovery for Associative Classification

6.2.1 Issues in Conventional Rule Discovery for Associative Classification

The first issue is the computational complexity of the rule discovery process. The problem lies in class association rule searching due to the huge search space of possible rules. When the support threshold is reduced, the number of generated candidate rules will be increased exponentially. This leads to a long runtime and huge usage of memory and disk space. This also means that there exists a support threshold lower bound, below which the support threshold may cause performance degradation of the rule discovery process.

As discussed in Section 5.1 of Chapter 5, large training datasets may achieve a better accuracy in associative classification in comparison with small training datasets. At the same time, the very small optimal support values corresponding to large datasets may lead to a more complex rule discovery process. If the optimal support value is below the support threshold lower bound, the best accuracy may never be achieved. Although Mstep-AC has achieved considerable improvements in comparison with conventional AC approaches in dealing with large datasets, it may still face the problem of increasing complexity when the optimal support threshold value is reduced.

The second issue is on the integration of the rule discovery and classification processes, which are two very different tasks. Most classification techniques have mechanisms to avoid overfitting/underfitting problems [Fre00]. However, overfitting/underfitting problems are generally neglected in association rule discovery for AC. The rule discovery process is simply to find association rules satisfying the support and confidence constraints regardless of whether or not the rules would be overfitting/underfitting the data. In addition, association rule mining is a well-defined, deterministic task which requires predefined support and confidence thresholds. All association rule mining algorithms discover exactly the same set of rules that contains all rules satisfying the specified support and confidence threshold values. In contrast, classification is an ill-defined, non-deterministic (or uncertain) task of predicting the class of new samples [Fre00]. The support and confidence threshold values for generating the best set of association rules for classification may vary depending on the nature of the dataset.

6.2.2 Association Rule Discovery for Associative Classification

To achieve an effective associative classification, it is not necessary to mine all the possible association rules, but only a subset of them that can be used to form an accurate classifier. That is, instead of using conventional association rule mining algorithms for rule discovery, we can search for a set of association rules applicable directly for the classification purpose.

Associative classification is based on a set of association rules, which are selected based mainly on the support and confidence measures. Apart from support and confidence, another factor that affects the accuracy of AC is the proportion of data samples from a dataset that can be classified by the set of rules. We define a measure called *coverage* to determine the proportion of the dataset that is covered by a set of rules. The coverage measure of a set of classification association rules S for a dataset D is the percentage of data samples in D, in which there exists at least a rule in S matching each data sample. The coverage measure is defined as follows:

$$cover(S, D) = \frac{|\{d \mid (d \in D) \wedge (\exists R : (R \in S) \wedge (R \ matches \ d))\}|}{|D|} \tag{6.1}$$

The concept of coverage has been used in [LHM98, LHP01] for selecting association rules for AC from the discovered rules. During the rule selection process, the set of discovered rules is first sorted in decreasing order of the confidence value. For each rule selected, the data samples that match it are removed from the training dataset [LHM98]. When the training dataset becomes empty, the set of selected rules will cover the whole training dataset, and the rule selection process is terminated. This helps select a small set of rules which covers the whole training dataset for classification. However, it is not effective to incorporate coverage into the rule discovery process of conventional AC approaches. As association rule mining is mainly based on frequency (i.e., the support measure) to search for association rules, the set of discovered rules may not be those with the highest confidence values if coverage is used to control the rule discovery process. As such, the associated classification process based on this set of rules may not be accurate.

In our proposed approach, we will use the coverage measure during the rule discovery process which helps gather a set of rules that covers the training dataset without containing too many redundant rules. As the three measures, namely support, confidence and coverage, can affect the quality of association rules mined for associative classification, they can be considered as *constraints* for mining a set of CARs for AC. The three constraints are summarized as follows:

$$supp(R) \geq minSupport \text{ for every } R \in S \tag{6.2}$$

$$conf(R) \geq minConfidence \text{ for every } R \in S \tag{6.3}$$

$$cover(S) \geq minCoverage \tag{6.4}$$

The confidence value of a rule is the probability that the rule predicts the class of a data sample matching it correctly. Thus, the best rule(s) that can be used to classify a data sample is the rule(s) with the highest confidence value in the set of rules that match it. In conventional AC, all rules satisfying the support and confidence measures are generated. It means that for most of the data samples, the optimal prediction with the highest confidence rule(s) is obtained. In our proposed approach, only a subset of rules satisfying the support and confidence thresholds is gathered to avoid massively searching the rule space. Therefore, it is important that the generated subset should contain most of the highest confidence rules so that the classification accuracy is comparable to conventional AC approaches. Thus, we need to increase the confidence threshold (i.e., *minConfidence*) as high as possible, as long as we can find a set of rules S satisfying the three constraints. Note that equation (6.3) is equivalent to:

$$min(\{conf(R) \mid R \in S\}) \geq minConfidence \tag{6.5}$$

Thus, our process on association rule discovery for AC becomes searching for a set of rules S such that S satisfies constraints (6.2) and (6.4), and the minimal confidence value of the rules in S is maximized.

6.2.3 AIS-based Model for Rule Discovery

In our proposed approach, the problem of mining a set of association rules for AC is considered as an optimization process for searching a set of association rules S. The optimization process searches for association rules in an evolutionary manner based on the artificial immune system, or in particular, the clonal selection principle. For mining association rules with AIS, association rules are considered as immune cells. The population in each generation is then considered as a set of association rules. The resultant association rules will be extracted from the memory pool. The processes of CLONALG shown in Figure 6.1 can be adapted to search for association rules as follows:

- *Affinity calculation.* Our goal is to search for rules with the highest confidence values. Thus, the confidence measure plays the role of affinity in AIS so that cells with high affinity are considered as good cells in the system. As such, the resultant memory pool contains high confidence rules which are important for achieving good accuracy for associative classification.

- *Rule selection.* Rule or cell selection is based on affinity, i.e., confidence measure. To ensure that the selected rules satisfy the support constraint, the support measure is used to filter out specific rules from the population before the rule selection and reselection processes.

- *Termination condition.* The coverage of the memory set will be increased during the evolutionary process as more rules are inserted into the memory pool after each generation. The search is completed when the coverage value reaches the minimum coverage threshold *minCoverage*. In other words, the coverage constraint can be considered as the termination condition of the evolutionary process.

It is clear that the set of association rules obtained from the memory set after the evolutionary process will satisfy the support and coverage constraints. The support constraint is held for every rule of the population so that rules in the memory pool also satisfy the constraint as they are selected from the population. The coverage constraint is held as it is the termination condition of the evolutionary process and the process is terminated when this condition is true.

In addition, the CLONALG algorithm, which was proposed to search for a population of maximal affinity cells, can be used to search for a set of rules with the highest confidence value that is effective for AC. The evolutionary model can be used to obtain the confidence threshold that is close to the optimal value. Besides, the CLONALG mechanism can help achieve a diverse number of local optimal solutions [dCVZ00a]. The diversity of rules means the diversity of itemsets of rules, which implies a higher number of data samples that the set of rules can match. As such, the proposed AIS-based model can gather a set of rules with high coverage values. It is, therefore, expected that the set of rules obtained by AIS-AC will be appropriate for AC. By using the evolutionary approach with appropriate constraints, we can avoid an exhaustive search of association rules. The computational complexity of AIS-AC is bounded by the population and the number of generations. This is probably much more stable compared with conventional AC approaches.

Moreover, the set of discovered rules can be evaluated after each generation of the evolutionary process. The result of the evaluation can then be used to refine the parameters such as the support threshold, population size, clonal rate, or search strategy so that a better set of rules for classification can be gathered in subsequent generations of evolution. In other words, the association rule discovery framework is adaptable towards the classification task.

6.3 AIS-AC Approach for Rule Discovery

In this section, we present an implementation of the proposed AIS-based model for rule discovery, which is called AIS-AC. In AIS-AC, the evolutionary process is carried out as follows. Firstly, in each generation, the support constraint is used to filter out specific rules from the population. Next, the confidence values of the rules are used for affinity selection. The population is cloned, mutated and diversified. Finally, the best rules in the population are moved to the memory based on the confidence constraint. At the end of each generation, the coverage measure is calculated based on the memory rules to decide whether the process should continue with another generation or be terminated. The process will be terminated when the coverage constraint is satisfied or the number of generations reaches a predefined maximum number of generations. If the process continues, the remaining population is input into the next generation.

The proposed AIS-AC algorithm, which is given in Algorithm 6.1, mines association rules for each class separately. The resultant memory set, which will be used for the classification process, includes all the memory sets of association rules according to all the classes in the dataset. As shown in Algorithm 6.1, AIS-AC follows the general framework of the clonal selection algorithm CLONALG, which includes the selection, cloning, mutation and diversity introduction processes. The population P is first initialized by the set of all 1-itemset rules $iset \Rightarrow c$ in which $iset$ contains a single item.

6.3.1 Selection

The selection process (in lines 4-9) eliminates rules with support values below the support threshold. In addition, rules of low confidence values in P will be eliminated, and only the first *selectionNumber* highest confidence rules are retained in P. In AIS-AC, the confidence measure is considered as affinity, choosing the first *selectionNumber* highest confidence rules will move the population towards containing higher affinity rules. The *selectionNumber*

parameter controls the number of populations for each generation during the evolutionary process.

Algorithm 6.1 AIS-AC for Rule Discovery in Associative Classification.

Input:

c - a class in the dataset for classification
I - the set of items in the dataset
minSupport - minimum support threshold value
minConfidence - minimum confidence threshold value
minCoverage - minimum coverage threshold value
noOfGenerations - maximum number of generations
selectionNumber - population after selection
clonalRate - clonal rate

Output:

M - memory set containing a set of association rules for AC

Process:

1: Initialize $P = \{\{item\} \Rightarrow c \mid item \in I\}$
2: Initialize $M = \emptyset$, $generationCount = 0$
3: **repeat**
4: **for all** rule R in P **do**
5: **if** $supp(R) < minSupport$ **then**
6: Remove R from P
7: **end if**
8: **end for**
9: Set P to *selectionNumber* highest confidence rules in P
10: Clone P with *clonalRate*
11: Mutate P
12: Diversify P
13: Count P
14: Prune P
15: **for all** rule R in P **do**
16: **if** $conf(R) \geq minConfidence$ **then**
17: Move R into M
18: **end if**
19: **end for**
20: $generationCount = generationCount + 1$
21: **until** $(cover(M) \geq minCoverage) \vee (generationCount \geq noOfGenerations)$
22: **return** M

6.3.2 Cloning

In the design of the clonal selection algorithm, the clonal rate of a cell is directly proportional to the affinity of the cell. It means that more offspring will be produced for higher affinity cells. Applying it to rule mining, if a rule R_1 has a higher confidence value than that of a rule R_2, then R_1 will produce more offspring than R_2. The search will focus more on

rules "around" R_1 as these rules potentially have a higher confidence value. However, in AC, it is generally not very effective to find many similar association rules. The reason is that similar rules usually match the same data samples, and in order to classify a test sample, one association rule is enough. It means that many of these "similar rules" are redundant.

In AIS-AC, we have selected a single *clonalRate* for every cell (in line 10), such as R_1 and R_2, to produce the same number of offspring. As such, we may not search every potentially high-confidence rule around R_1, but instead we give more chances to find neighboring rules of R_2, which may help increase the diversity of the set of the resultant rules. In other words, this may slow down the process of moving towards local optimal cells slightly, but it helps keep the population more diverse.

6.3.3 Maturation

In the clonal selection algorithm, the mutation rate of a cell is inversely proportional to the affinity of the cell. It gives low affinity cells a chance to "mutate" more in order to improve its affinity. In AIS-AC, the mutation rate is equal to "one item" for every rule (in line 11). That is, when a rule is mutated, the newly produced rules will differ from the parent rule only by one item. The reason is that cells are rules with relatively small numbers of items (mostly 2-5). Two rules with two or more different numbers of items may mean different implications or have no relation at all. For example, let's consider an association rule R: {*bread, milk, cheese*} $\Rightarrow$ *breakfast*. The itemsets with 1-item different from {*bread, milk, cheese*}, such as {*bread, milk*}, {*bread, milk, egg*} or {*bread, milk, cheese, potato*}, may still show that they are items for breakfast. However, itemsets with two or more items different from {*bread, milk, cheese*}, such as {*bread, egg, beef*}, {*cheese, potato, hamburger*} may suggest that they are items for lunch or dinner. The restriction based on minimal (one item) difference keeps the "inheritance" trait among the generations.

6.3.4 Diversity Introduction

During the selection process, with the high affinity selection, the population tends to move towards some local optimal regions. To maintain diversity, the clonal selection algorithm updates the population by replacing some existing cells with new ones. For the very large search space of possible rules, if we generate new rules arbitrarily, the quality of the rules (based on support and confidence) would be very low. These rules are likely to be filtered out in the next selection step, thereby playing no role in the overall process at all.

In AIS-AC, instead of rule replacement, new rules are generated and added into every generation (in line 12). To gather better rules, we have a separate process for executing the selection, clonal and maturation steps of AIS-AC. This separate process synchronizes with the main AIS-AC process and aims to generate new rules after a specified number of generations (say $n = 3$) from the initial 1-item rules. This mechanism brings much higher confidence values for new generated rules. After the first n generations in AIS-AC, new rules can be generated for AIS-AC to help diversify the population of P. This is an important and effective step to help improve the quality of the mined association rules.

6.3.5 Support and Confidence Counting

In conventional AC approaches, an association rule mining algorithm such as Apriori [AS94] is used to massively search for the set of all possible association rules based on the support and confidence constraints. In the AIS-AC approach, it only mines a subset of CARs that satisfies the support, confidence and coverage constraints. In the current implementation of AIS-AC, we have adopted a hash-tree structure used in the Apriori algorithm for fast support and confidence counting. The support and confidence counting process is carried out (in line 13) for rules in both the population and the new rule generation process for diversity introduction. This means that AIS-AC conducts a single scan on the dataset for a generation.

6.3.6 Population Pruning

The population pruning process (in line 14) prunes those rules that have already been covered by the memory rules from the population of rules. Given two rules $R : iset \Rightarrow c$ and $R' : iset' \Rightarrow c$, if R is more general than R' (i.e., $iset \subseteq iset'$) and the confidence value of R is higher than that of R', then R' is covered by R. Therefore, if R' is covered by R, then the exploration around R' will probably result rules that are also covered by R. As such, when R is a memory rule, the discovery of rules covered by R is not necessary. This is explained as follows. Since R is more general than R', if a transaction t matches R', then it will match R (i.e., $iset \subseteq iset'$ and $iset' \subseteq t$, then $iset \subseteq t$). Thus, when R is a memory rule, R' does not help increase the coverage of the memory set. In addition, since the confidence value of R' is less than that of R, if a transaction matches both R and R', then it should be classified using R. Thus, R' does not help increase the classification accuracy. The pruning process helps the searching focus on other regions in the rule space that are not covered by the current memory rules.

6.3.7 Memory Selection

The memory selection process (in lines 15-19) adds high confidence rules into the memory pool. As the optimal (maximal) value of confidence threshold may vary depending on the dataset, there is no fixed value for the optimal *minConfidence* that can be used to select rules from the population into the memory pool. In our approach, rules are selected from the population with the confidence threshold *minConfidence* initially set to a high value. The confidence threshold is then subsequently reduced after a certain number of generations until the found rules can cover the dataset.

6.3.8 AIS-AC Approach and Association Rule Mining Principle

In conventional association rule mining algorithms, the support constraint with anti-monotone property is used sufficiently for reducing the combinatorial search space. One example is the generate and count framework which generates only the itemsets of which subsets are frequent for support counting. In an evolutionary search, this property is exploited as follows. Let $iset$ be a frequent itemset and $iset \Rightarrow c$ be a rule satisfying the support and confidence constraints. It can imply that if an itemset sub_iset is a subset of $iset$ ($sub\_iset \subseteq iset$), then sub_iset is a frequent itemset. It can also imply that if itemset sup_iset is a superset of $iset$ ($iset \subseteq sup\_iset$), then sup_iset is potentially a frequent itemset as it contains the frequent subset $iset$. In addition, as sub_iset and sup_iset have many items in common with $iset$ (in the AIS-AC algorithm presented above, sub_iset and sup_iset are only 1-item different from $iset$), the confidence values presenting the implications $sub\_iset \Rightarrow c$ and $sup\_iset \Rightarrow c$ probably fluctuate around that of the implication $iset \Rightarrow c$ which is high enough to satisfy the confidence constraint. Thus, we may consider $sub\_iset \Rightarrow c$ and $sup\_iset \Rightarrow c$ as candidate rules which should be counted based on the support and confidence constraints.

We also observe that sub_iset and sup_iset may be obtained when we impose mutation operator on $iset$ as follows. Let's consider one of the simplest representations of itemsets - the binary representation. An itemset $iset$ is represented by a binary string of N bits where N is the number of the set of items. For $i = 1$ to N, if i^{th} bit of $iset$ is set to 1, then the i^{th} item is included in $iset$; otherwise the i^{th} item is not included in $iset$. For example, let {1,2,3,4} be a set of items. The string "0101" then represents itemset {2,4}. Now, consider a mutation operation on "0101". When the operator turns a bit "0" into "1", we have a superset of {2,4} such as {1,2,4} (i.e., "1101") or {2,3,4} (i.e., "0111"). In contrast, when the operator turns a bit "1" into "0", we have a subset of {2,4} such as {2} (i.e., "0100")

or {4} (i.e., "0001"). In summary, we may say that the mutation (or maturation) operator also exploits the association rule mining principle for candidate expansion in an evolutionary search.

6.4 Performance Evaluation

The performance of the proposed AIS-AC approach is evaluated using a 3.0 GHz Pentium PC with 1 GB of memory running with the Microsoft Windows XP. We have implemented the AIS-AC algorithm and the msCBA algorithm [LHM98, LHP01, LMW01] for comparison. The msCBA algorithm is based on the Apriori algorithm [AS94] for association rule mining.

Three experiments have been conducted. The first two experiments have been conducted for evaluating the performance of the proposed AIS-AC approach in comparison with msCBA. Both algorithms are studied with a large range of support threshold values. In the implementation of the AIS-AC algorithm, the population was set to 100 and the maximum number of generations was set to 50. The clonal rate and the number of new rules for diversity introduction were both set to 20. The coverage threshold was set to 98%. In the third experiment, the AIS-AC is evaluated with various values of AIS parameters. The aim of this experiment is to examine the effects of different parameter values on the performance of AIS-AC.

The first and third experiments use the four attribute datasets, namely *Adult*, *Digit*, *Letter* and *Nursery*, while the second experiment uses a textual dataset, namely *Reuters-R8*, for performance evaluation. The attribute and textual datasets have been described in Section 5.3.1 of Chapter 5 and Section 4.2.4 of Chapter 4 respectively.

6.4.1 Experimental Results for Attribute Datasets

Figure 6.2 shows the performance results of the first experiment for the two approaches using the attribute datasets based on accuracy (in bar chart) and runtime (in line chart). The experiment was conducted with the total minimal support threshold ranging from 0.3% to 10%. The performance of AIS-AC was obtained from 10 runs for each dataset and support thresholds. The average accuracy is shown with the standard deviation, which is relatively small and mostly less than 0.6%.

From the performance results, we have observed that the runtime of AIS-AC is quite steady with different support threshold values. In contrast, the runtime of msCBA has increased exponentially when the support threshold is decreased. Further, the performance of msCBA with small support values is not shown in Figure 6.2 as we are unable to obtain the

performance results for corresponding support values. This may be due to the reason that the runtime was too long or the number of possible rules was too large so that the system ran out of memory during the rule mining process. The ability of mining rules with small support values gives AIS-AC an advantage to deal with large datasets. For example, the real-world Adult dataset may contain data on tens or hundreds of millions of people. For this kind of data, a support of 0.1% may easily refer to data for tens or hundreds of thousands of people, which may represent patterns for certain groups of people based on race or profession. In such cases, rules with very small support values are very important and should be discovered.

Let's consider the performance in terms of accuracy for both classifiers. The performance results have shown that AIS-AC performs comparably to msCBA for all four datasets with most of the threshold values. This means that the sets of class association rules discovered by AIS-AC are comparable to that discovered by msCBA for associative classification. In terms of best accuracy achieved in each dataset for both classifiers, the performance of the *Adult* dataset has shown that AIS-AC has outperformed msCBA with shorter runtime. AIS-AC has achieved the best accuracy of 85.8% with the support value of 0.3%, whereas msCBA has achieved the best accuracy of 85.6% with the support value of 0.6%. In the *Digit* and *Letter* datasets, msCBA has performed better than AIS-AC but with longer runtime. In the dataset *Nursery*, the msCBA algorithm has performed better and faster than AIS-AC. This is mainly due to the reason that the *Nursery* dataset contains only a rather small set of 27 items compared with more than a hundred items in the other datasets. As such, the number of combinations of items is relatively small. Therefore, in this small search space of association rules from this dataset, the AIS-AC algorithm loses its advantages.

6.4.2 Experimental Results for Textual Dataset

Figure 6.3 shows the performance results of the second experiment of the two approaches, AIS-AC and msCBA, using the textual dataset. The experiment was conducted with the total minimal support threshold ranging from 20% down to 0.6%. Similar to the performance results from the attribute datasets, the performance of msCBA with small support value of 0.6% is not shown as we are unable to obtain the result. Again, this may be due to the reason that the system might run out of memory during the rule mining process.

In general, the msCBA algorithm obtains the complete set of association rules (with the constraints on support and confidence). Therefore, it should be more accurate than AIS-AC

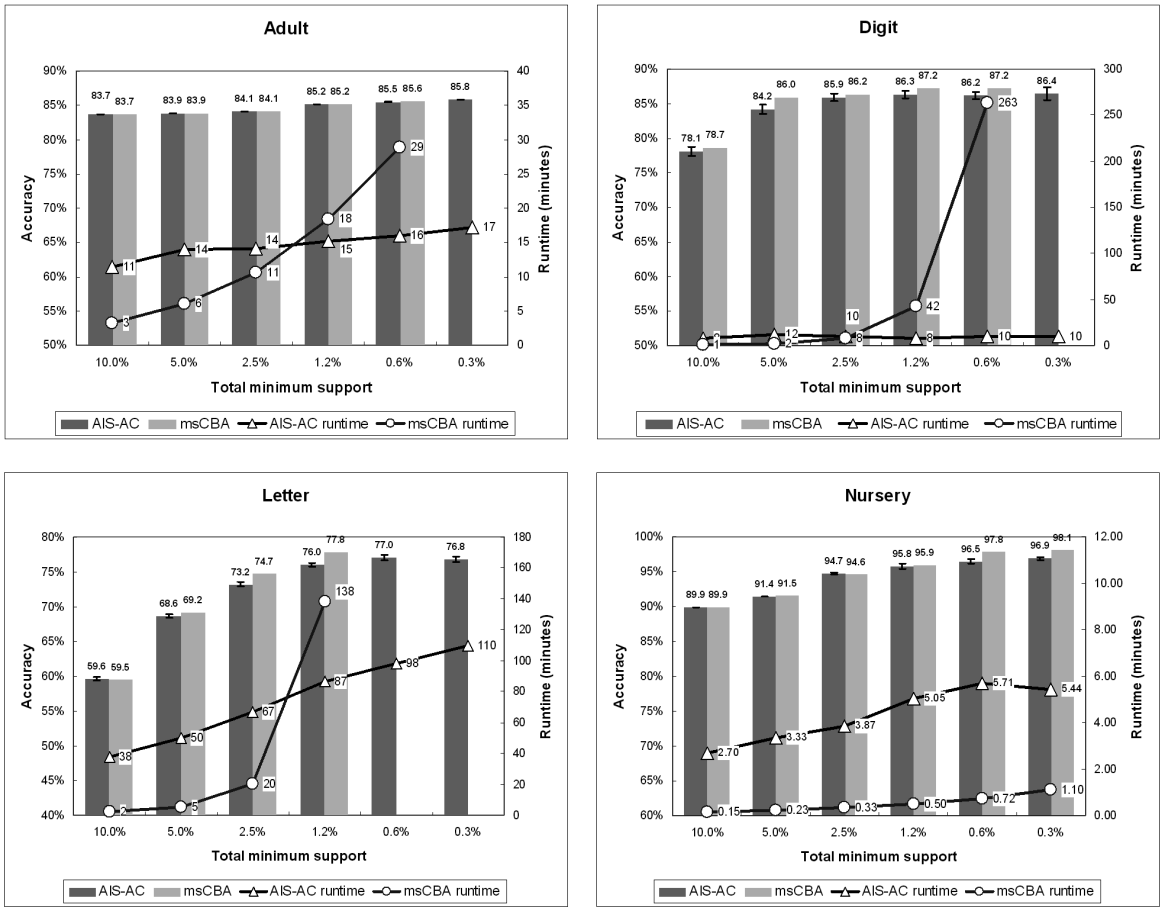


Figure 6.2: Performance results on four attribute datasets.

with the same support threshold value. However, the performance results from the *Reuters-R8* dataset have shown that AIS-AC has outperformed msCBA with most of the support threshold values. The possible reason is that AIS-AC mines data with the coverage constraint. For each class, the rule mining process stops when the coverage value reaches the threshold. This brings a balance of rules for each class.

We have also observed that the runtime of msCBA has increased exponentially when the support threshold is decreased whereas the runtime of AIS-AC is much more stable. Again, the performance results have shown the ability of the AIS-AC approach in mining association rules with small support values. This is especially useful for web mining, which deals with an exponential growth of web documents made available on Internet everyday.

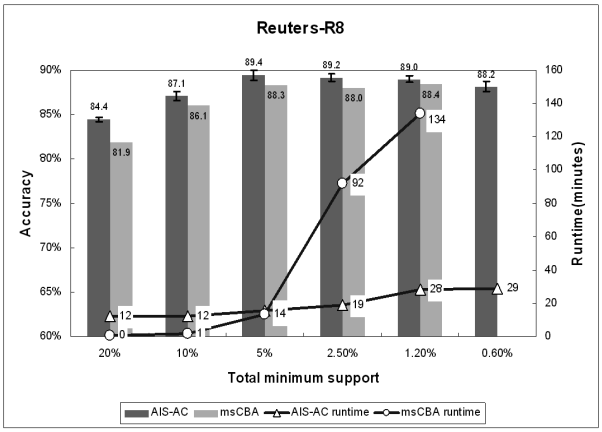


Figure 6.3: Performance results on Reuters-R8 dataset.

6.4.3 Experimental Results based on Different Parameters of AIS-AC

The third experiment has been carried out to evaluate the effects of different parameter values on the performance of AIS-AC. The parameters examined including the conventional parameters of AIS such as population, clonal rate and number of new rules for diversity introduction. Moreover, we also study an additional parameter, the coverage threshold value, that is used in AIS-AC. As there are a number of parameters, we carry out the experiment with the parameters being considered independently. That is, only a single parameter is examined at a time while setting other parameters with the values used in the first two experiments.

Table 6.1 shows the performance results of AIS-AC with different values on population. We can observe that larger populations help increase the performance of AIS-AC. This can be explained that a larger number of population would correspond to a broader region that the searching process can explore. That is, a better set of rules can be obtained which may help build a more accurate associative classifier. Obviously, a larger population probably takes more time to count the support and confidence of rules in each generation. The result is that the runtime of the searching process would be increased.

The performance results of AIS-AC with different clonal rates are shown in Table 6.2. Here, a low clonal rate value would improve the diversity of the search but may slow down the process of obtaining the optima of local regions. On the other hand, a high value of clonal rate may lead to a fast convergence in searching the optima from local regions, but the resultant set of rules can be less diversified. The results from Table 6.2 have shown that AIS-AC has achieved the best performance with clonal rates of 20 and 50.

Table 6.1: AIS-AC performance with different values of population.

Dataset	Population				
	30	50	100	200	500
Adult	85.2%	85.2%	85.2%	85.2%	85.2%
Digit	85.2%	85.5%	86.3%	86.4%	86.3%
Letter	73.9%	75.1%	76.0%	76.2%	77.3%
Nursery	95.3%	95.4%	95.8%	95.8%	95.9%
Average	84.9%	85.3%	85.8%	85.9%	86.2%

Table 6.2: AIS-AC performance with different values of clonal rates.

Dataset	Clonal rate				
	5	10	20	50	100
Adult	85.2%	85.2%	85.2%	85.2%	85.2%
Digit	85.7%	85.8%	86.3%	86.3%	85.8%
Letter	72.3%	75.3%	76.0%	75.7%	75.5%
Nursery	95.2%	95.5%	95.8%	95.5%	95.5%
Average	84.6%	85.5%	85.8%	85.7%	85.5%

We have also conducted the experiment with different values of the maximal number of generations. However, when the maximal number of generations increases above 50, the performance of AIS-AC does not change much. This is because most of the searching process for rules of different classes are terminated before the 50 generations of revolution (i.e., the coverage reaches 98%). Table 6.3 shows the performance results of AIS-AC with different coverage threshold values. To ensure the searching process reaches a coverage threshold value of above 98%, the maximal value of generations is set to 100. We may observe that the performance is worsened with the coverage threshold values of 90% and 100%. This can be explained that with a small coverage threshold, many data samples may not be covered by any rules which may result in underfitting learning. On the other hand, with a very large coverage value (such as 100%), the algorithm may learn exhaustively which may produce noise and cause overfitting.

The performance results of AIS-AC with different numbers of new rules for diversity introduction are shown in Table 6.4. If this number gets smaller, the searching process may be less diversify. Instead, the searching process focuses on searching only a small number of local regions, thereby obtaining the optima quickly. When the number of new rules for

Table 6.3: AIS-AC performance with different values of coverage thereshold.

Dataset	Coverage threshold				
	90%	95%	98%	99%	100%
Adult	85.1%	85.1%	85.2%	85.2%	85.2%
Digit	85.7%	86.1%	86.3%	86.3%	85.8%
Letter	72.3%	75.5%	76.0%	75.9%	75.5%
Nursery	95.2%	95.5%	95.8%	95.6%	95.5%
Average	84.6%	85.6%	85.8%	85.7%	85.5%

Table 6.4: AIS-AC performance with different numbers of new rules for diversity introduction.

Dataset	No of new rules / the population				
	1/20	1/10	1/5	1/3	1/2
Adult	85.1%	85.1%	85.2%	85.2%	85.2%
Digit	86.3%	86.4%	86.3%	86.4%	86.3%
Letter	75.3%	75.7%	76.0%	75.7%	75.6%
Nursery	95.8%	95.6%	95.8%	95.6%	95.5%
Average	85.6%	85.7%	85.8%	85.7%	85.6%

diversity introduction gets larger, the searching process can be more diversified. However, the process may converge slower to the optima for larger searching regions. The results from Table 6.4 show that AIS-AC has achieved the best performance with the number of new rules for diversity introduction equal to 1/5 of the population.

6.5 Summary

In this chapter, we have proposed an approach called AIS-AC, which is based on the clonal selection theory of artificial immune systems, for searching association rules for AC. Following the population-based search model of evolutionary algorithms, AIS-AC has the capability of dealing with complex search space of association rules. The performance results have shown that the proposed AIS-AC approach has achieved good runtime and accuracy in comparison with a conventional AC approach.

Moreover, in this approach, the set of discovered association rules can be evaluated for classification purpose after each generation of evolution. Consequently, the searching criteria

can be adjusted dynamically in subsequent generations according to the classification evaluation. The integration of the rule discovery and classification processes is, therefore, naturally feasible.

Chapter 7

Conclusions

In this chapter, we first summarize the work that has been done in this research. Then, we give the directions for further research work.

7.1 Summary

Association rule mining is one of the major data mining techniques, which aims to generate rules in the form of $X \Rightarrow Y$, where X and Y are sets of items in a given database D and $X \cap Y = \emptyset$. The mining process of association rules is based on the measures of support and confidence so that only "useful" and "certain" rules are gathered. One of the challenges in association rule mining is that the number of rules mined under the support-confidence framework is often too large and many of them are uninteresting. In addition, the association rule mining process is computationally expensive as it requires exhaustive search in the huge space of possible rules.

Association rules can be used for data classification. Associative classification makes use of association rule mining in discovering accurate class association rules for classification. However, similar to association rule mining, one of the main problems of using association rule mining in AC is the very large search space of possible rules. This may cause performance degradation in the rule discovery process, especially with small support threshold values which are very important for building accurate associative classifiers from large training datasets. In addition, the set of association rules mined using the association rule mining framework may not be effective to use directly in the classification task.

This research aims to investigate techniques to tackle these problems on association rule mining and rule discovery for associative classification. The contributions of this research are listed as follows:

- A new type of constraint for association rules called category-based constraint has been proposed. The category-based constraint stipulates the category patterns of itemsets. Thus, this constraint can be used with datasets, in which the data is categorized. In addition, an algorithm for association rule mining with the category-based constraint called Apriori^{CB} has been developed based on the Apriori algorithm. This approach is efficient because, by using the category patterns, it can reduce considerably the number of itemsets to be processed in each pass of the Apriori algorithm.

- Effective measures for association rule mining have been assessed and analyzed. In particular, the relative confidence, which is captured by the certainty factor measure, is able to represent the characteristics of the implication relationships of itemsets of rules. Relative confidence can be used instead of the confidence measure to overcome drawbacks on generating many misleading rules in the conventional support and confidence framework.

- A new approach called associative feature selection has been proposed for unsupervised feature selection from textual datasets. The proposed approach measures the importance of terms based on the associative strength of terms among documents from a document collection. The proposed unsupervised associative feature selection approach can be used effectively for data preprocessing of textual datasets which are often large and contain a lot of irrelevant features.

- A multiple-step rule discovery approach for AC called Mstep-AC has been proposed. In this approach, class association rules are obtained by performing association rule mining repeatedly for a specified number of steps. The repeated steps focus on discovering rules on those data samples that might possibly be misclassified. The proposed approach has achieved better associative classification accuracy while maintaining the rule discovery process to the level of complexity comparable with conventional AC approaches.

- An Artificial Immune System based approach called AIS-AC has been proposed for discovering class association rules from large datasets. Instead of massively searching for all possible rules, AIS-AC will only find a subset of rules that is suitable for effective

AC in an evolutionary manner. The proposed approach is efficient in dealing with the complex search problem of association rules in rule discovery for AC.

7.2 Future Work

In this research, effective techniques have been developed to enhance association rule mining and associative classification. Nevertheless, this research can be further extended in the following directions: investigating associative feature extraction for text mining, extending the current rule discovery approaches for AC, investigating rule selection for AC and applying the proposed AC techniques for web mining.

7.2.1 Associative Feature Extraction for Text Mining

In this research, we have proposed the associative feature selection approach for text mining. Apart from feature selection, feature extraction approaches such as Principal Component Analysis (PCA) [Jol86] and Random Projection [San04] are also frequently used for text mining [LMY03]. While feature selection selects a subset of features from the original set, feature extraction creates a smaller set of new features from the original features. Because the results of feature selection and extraction are a smaller set of features compared to the original set, these processes are usually considered as feature reduction or dimensionality reduction [LM98] which is very important for solving the problem of high dimensionality of feature space in text mining.

For future work, we can investigate association rule based feature extraction approach that makes use of association rules for extracting features from textual documents. In associative feature selection, relations among the features represented by association rules are used to score the relevance of these features to the textual domain. It means that their relations contain information about the features in the database. Therefore, the relations represented by association rules also can be used to create new features. In other words, association rule mining can be used for feature extraction that extracts new features from textual documents. Similar to associative feature selection, associative feature extraction can take into account the advantage of association rules in representing the relations between items. Associative feature extraction is an unsupervised process as association rule mining does not require any labelled data. Thus, it can save the cost of labelling the data compared with the supervised process. This is very important for text mining in which we often need to process a huge number of textual documents.

7.2.2 Extending the Mstep-AC Approach for Associative Classification

To achieve a better AC accuracy (i.e., reducing the error rate), the proposed Mstep-AC approach focuses on discovering rules for the low-ranked data samples in which classification errors are most likely to occur. To do this, Mstep-AC performs the rule discovery process repeatedly for a specified number of steps. At the end of each step the high-ranked data samples are removed from the dataset by a fixed dataset reduction rate so that only low-ranked data samples are retained for rule discovery in the next step. The discovery process in each step is carried out using the popular breadth-first search based Apriori algorithm. Although the experimental results given in Chapter 5 have shown that the Mstep-AC approach has improved the accuracy of AC considerably, the current implementation of Mstep-AC may not be the optimal one. And the performance of the Mstep-AC approach can further be improved. For example, the determination of the most appropriate dataset reduction rate for each step may help achieve better set of rules for classification, or the use of another effective rule discovery approach in each step may give a better AC performance in terms of runtime and memory usage.

For future work, we can investigate an effective implementation for the Mstep-AC approach. In particular, appropriate dataset division strategies need to be studied. For example, as the size of the dataset decreases after each step of the multiple-step rule discovery process, the rule discovery process in the earlier steps tends to be more complex while the rules mined from the later steps are more likely to overfit the data. Thus, if we set the dataset reduction rate to a large value in the earlier steps and a small value in the later steps, a simpler multiple-rule discovery process with less overfitting rules can be achieved. In addition, we can also apply depth-first search based algorithms such as Eclat and FP-growth to discover CARs in the rule discovery process in each step of Mstep-AC. The fast support and confidence counting mechanisms of Eclat and FP-growth can help speed up the overall multiple-step rule discovery process.

7.2.3 Extending the AIS-AC Approach for Associative Classification

The AIS-AC approach searches for class association rules based on Artificial Immune System in an evolutionary manner. In each generation, the approach uses the hash-tree structure of the Apriori algorithm for support and confidence counting. Similar to the Apriori algorithm for association rule mining, AIS-AC spends most of its runtime on counting the support and confidence values of rules. The counting operation slows down the rule discovery process as

well as limits the size of the population and the number of generations in the evolutionary process.

For future work, we can apply other support and confidence counting mechanisms for fast support and confidence counting. For example, the counting process for each generation can be faster with the use of vertical database layout or compressed FP-tree structure. This will help increase the accuracy of the classification task as we can increase the number of individuals in the population as well as the number of generations, which will lead to a better set of association rules for AC.

In addition, we can also apply the strategy of searching for more rules for low-ranked data samples of Mstep-AC to the evolutionary searching process of AIS-AC. That is, during the evolutionary searching process of AIS-AC, high-ranked data samples are removed from the training dataset so that subsequent generations of AIS-AC will focus on finding only rules for low-ranked data samples. Similar to the Mstep-AC approach, this can help achieve a better accuracy for the AIS-AC approach.

Moreover, in AIS-AC, the constraints of the targeted rule set are formulated as an optimization problem. This enables the use of soft computing approaches, which are tolerant of imprecision and approximation, for the rule discovery process. Therefore, for future work, we can also investigate other nature-inspired evolutionary optimization algorithms such as Genetic Algorithm [Hol92] and Particle Swarm [KE95] to deal with a large search space for association rule discovery based on our proposed AIS-AC framework with minor modifications.

7.2.4 Rule Selection for Associative Classification

Generally, rule selection consists of two processes, namely rule ranking and subset selection. The rule ranking process aims to rank rules in the correct order of classification accuracy so that the best rule can be chosen for classifying each data sample. In AC, the rule ranking process is based mainly on the confidence measure. If the confidence values of two rules are the same, then the support measure is used to decide the better one [2]. One of the challenges here is how to evaluate (i.e., count) the support and confidence values that truthfully reflect the classification accuracy of the rules. In the rule discovery process of AC, the support and confidence values can be considered as *global evaluation* as they are counted using the whole training dataset [LHM98]. In contrast, the *covering approach* [Mic80], which searches for rules one by one, is regarded as *local evaluation*. The approach first finds the best rule

and then removes those training data samples matching that rule. The process is then repeated recursively with the remaining data samples. As the data samples matching a rule are removed from the training dataset when the rule is found, subsequent rules are searched and evaluated locally using only the data samples that do not match any currently found rules. It is reported in [LHM98] that the global evaluation approach outperforms the local approach in rule ranking for AC. However, both approaches have their drawbacks. The local evaluation approach, which evaluates rules using the exact samples they are probably used for classification, tends to overfit the data [LHM98]. On the other hand, the global evaluation approach is likely to underfit the data as it uses a large set of data samples (i.e., the whole dataset) to evaluate a rule that may be applicable to only a small set of data samples (i.e., the data samples which are probably classified by that rule).

For future work, we can investigate a ranking approach that can deal with both overfitting and underfitting problems of rule evaluation. One of possible solutions is to specify an appropriate range of data samples for evaluation of each rule. The evaluation range of a rule should not be too specific so that it can avoid the overfitting problem as in the covering approaches. On the other hand, the evaluation range of a rule should also not be too general so that it can avoid the underfitting problem as in conventional AC approaches. Another solution is to apply probabilistic approaches for a more precise evaluation. For example, to evaluate a rule, we may compute an appropriate weight for each data sample that presents the probability in which the rule is used for classifying the data sample. The support and confidence of the rule are then counted using the whole dataset of data samples with corresponding weights. That is, if a data sample is probably classified by the rule, the contribution (or weight) of it on the support and confidence of the rule will be high and vice versa. The use of the whole dataset for rule evaluation may help avoid the overfitting problem, while the focus on assigning higher weights to data samples that are probably classified by the rule for evaluating it may help avoid the underfitting problem.

Another challenge related to the ranking problem is how to use the support and confidence measures for estimating the classification accuracy of the class association rules. The confidence measure of a rule which is the classification accuracy of the rule on the training data can be used to estimate precisely the classification accuracy of the rule on new test data when the support value is large enough. However, as discussed in Section 5.1.4 of Chapter 5, rules with small support values can also be used to improve the accuracy of AC. In this case, the estimation of classification accuracy using the confidence measure, which is based

on a small number of data samples, may not be accurate. To estimate the classification accuracy or *expected accuracy* [CB91] of a rule, one may use *Laplace expected accuracy estimate* [Nib87] or *m-probability-estimate* [Ces90]. Basically, these measures can be calculated from the support and confidence measures, and their values are close to the confidence value for a rule with a large support value. These measures, however, are used to estimate the expected accuracy of rules independently. In [DHF05], we have developed a measure called *prediction confidence*, which aims to estimate the classification accuracy of rules for AC. The proposed prediction accuracy, however, is computed based on the assumptions that the discovered rules are independent and distributed evenly on the dataset [DHF05]. This may not be always true for real-world datasets. For future work, we can investigate appropriate techniques for computing these factors (i.e., the distribution of rules and their independence of each other) from the set of discovered rules and the dataset. The prediction confidence will then present the classification accuracy of class association rules for AC more precisely. It means that better performance of AC in terms of accuracy can be achieved.

In addition to rule ranking, we can also investigate the problem on subset selection which aims to select the best subset of rules for classification from the set of discovered rules. The huge search space of possible subsets from the set of discovered rules makes it almost impractical to evaluate exhaustively all possible subsets in order to find the best one. Currently, the AC approaches use a heuristic approach that considers rules that are ordered. That is, a higher ranked rule always takes a higher priority. For future work, we can investigate a more effective subset selection approach. The approach should be able to find a subset of discovered rules for classification, which is not limited to the ranked order of rules. Similar to association rule mining, pruning techniques may help reduce the number of subsets for evaluation. For example, similar to the support property in association rule mining, some subsets may exclude others so that the number of subsets to be considered could be reduced. In addition, an approach for fast evaluation of a subset of discovered rules for classification is also important for the subset selection process. Fast subset evaluation can help evaluate more number of subsets so that the best one can be obtained. One of the possible approaches is to use a tree structure to store the dataset in the system memory which can help achieve fast subset evaluation [Li01].

7.2.5 Associative Classification for Web Mining

In web mining, classification of web documents into predefined classes is an important task, especially with the exponential growth of the number of web documents available on the Internet today. Web classification can be used for many applications. For example, web directory service systems such as Yahoo and Google News can automatically gather and classify web documents or information to form web data directories [HC02, KJ04]. Web document classification can also be used for web filtering [LHF02, Pea, Sur] through blocking prohibited web accesses.

In [DCH04], we have proposed a web monitoring and filtering system using web document classification. The proposed approach uses web mining techniques to classify web documents into predefined classes such as pornography, games, sports, entertainment, etc. The generated classes of documents are then used by the filtering system for blocking inappropriate web accesses. In this system, an conventional AC approach is applied for web classification based mainly on textual contents of web documents. The advantage is that once the associative classifier is constructed, the classification process is workable because web documents are classified based on a small set of selected class association rules. And a large training dataset of web documents is no longer needed in the classification process. Thus, conventional AC approaches can be used for efficient online web classification. In addition, selected class association rules are readable and understandable that can be easily modified by humans. In [AZ02], Antonie and Zaiane have also shown that a small number of class association rules can be added by domain experts to improve the accuracy of the AC system. However, conventional AC approaches may not be very effective for classification of a large set of web documents. As mentioned in Chapter 5, the rule discovery process of AC may be very complex, especially with small support threshold values which are very important for building accurate associative classifiers from large training datasets. Moreover, similar to AC for textual datasets, the rule discovery process on web documents may return an uneven set of discovered rules, which may cause performance degradation. In [DCH04], it has shown that conventional AC approaches are less accurate in comparison with Kohonen's Self-Organizing Maps (SOM) [Koh95] and Fuzzy Art [CGR91] for the selected set of experimental web documents.

For future work, we can apply the proposed Mstep-AC and AIS-AC approaches for web classification. As discussed in Chapter 5, the Mstep-AC approach can improve the classification accuracy while the AIS-AC approach can improve the classification efficiency of AC

in classification of textual datasets. Thus, Mstep-AC and AIS-AC can be used to improve the accuracy and efficiency of AC for web documents based on textual contents. In addition, we can further improve the performance of web classification by incorporating some structural features of web data such as HTML tags and URL links for AC [Fur99]. Furthermore, we can also investigate techniques for incremental update [TLC99] of class association rules in rule discovery for AC. This would be useful for supporting accurate classification of web documents due to the emergence of new web documents on the Internet everyday.

Appendix A

List of Publications

The work reported in this thesis has been published in the following international journals and conferences.

A.1 International Journal Papers

1. T.D. Do, S.C. Hui, and A.C.M. Fong. Associative Feature Selection for Text Mining. *International Journal of Information Technology (Revised Selected Papers from ICIC 2005)*, 12(4):59-68, 2006.
2. T.D. Do, S.C. Hui, and A.C.M. Fong. Associative Classification with Prediction Confidence. *Lecture Notes in Artificial Intelligence (Revised Selected Papers from ICMLC 2005)*, D. S. Yeung, Z. Q. Liu, X. Z. Wang, and H. Yan, editors, LNAI 3930:119-208, 2006.

A.2 International Conference Papers

1. T.D. Do, S.C. Hui, and A.C.M. Fong. Mining Frequent Itemsets with Category-based Constraints. In *DS'03: Proceeding of the 6th International Conference on Discovery Science*, October 2003, Sapporo, Japan. LNAI 2843, pp. 76-86, Springer.
2. T.D. Do, S.C. Hui, and A.C.M. Fong. Mining Association Rules using Relative Confidence. In *IDEAL'04: Proceeding of the fifth International Conference on Intelligent Data Engineering and Automated Learning*, August 2004, UK. LNCS 3177, pp. 306-313, Springer.

3. T.D. Do, K. Chang, and S.C. Hui. Web Mining for Cyber Monitoring and Filtering. In *CIS'04: Proceeding of the IEEE Conference on Cybernetics and Intelligent Systems*, pp. 399-404, December 2004, Singapore.
4. T.D. Do, S.C. Hui, and A.C.M. Fong. Artificial Immune System for Associative Classification. In *ICNC'05: Proceeding of the first International Conference on Natural Computation*, August 2005, Changsha, China. LNCS 3611, pp. 849-858, Springer.
5. T.D. Do, S.C. Hui, and A.C.M. Fong, Prediction Confidence for Associative Classification, In *ICMLC2005: Proceeding of the fourth International Conference on Machine Learning and Cybernetics*, pp. 1331-1340, August 2005, Guangzhou, China.
6. T.D. Do, S.C. Hui, and A.C.M. Fong, Associative Feature Selection for Text Mining, In *ICIC'2005: Proceeding of the International Conference on Intelligent Computing*, August 2005, Hefei, China,
7. T.D. Do, S.C. Hui, and A.C.M. Fong. Mining Class Association Rules with Artificial Immune System. In *KES2005: Proceeding of the Ninth International Conference on Knowledge-Based Intelligent Information & Engineering Systems*, September 2005, Melbourne, Australia. LNCS 3684, pp. 94-100, Springer.

Bibliography

[Ada01] J. M. Adamo. *Data Mining for Association Rules and Sequential Patterns*. Springer, Berlin, 2001.

[AE99] K. Aas and L. Eikvil. Text categorisation: A survey. Technical Report 941, Norwegian Computing Center, June 1999.

[AEMT00] K. M. Ahmed, N. M. El-Makky, and Y. Taha. A note on "beyond market baskets: Generalizing association rules to correlations". *SIGKDD Explorations*, 1(2):46–48, 2000.

[AFK97] A. Amir, R. Feldman, and R. Kashi. A new and versatile method for association generation. In *First European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'97)*, pages 221–231, London, UK, 1997. Springer-Verlag.

[AIS93] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D. C., USA, 1993.

[AS94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *20th International Conference on Very Large Data Bases (VLDB)*, pages 487–499, San Francisco, CA, USA, 1994.

[AY98] C. C. Aggarwal and P. S. Yu. A new framework for itemset generation. In *the 17th ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 18–24, Seattle, Washington, USA, 1998.

[AZ02] M. L. Antonie and O. R. Zaiane. Text document categorization by term association. In *IEEE 2002 International Conference on Data Mining (ICDM'2002)*, pages 19–26, Maebashi City, Japan, 2002.

[BBSV02] F. Berzal, I. Blanco, D. Sánchez, and M.A. Vila. Measuring the accuracy and interest of association rules: A new framework. *Intell. Data Anal.*, 6(3):221–235, 2002.

[BCG04] E. Baralis, S. Chiusano, and P. Garza. On support thresholds in associative classification. In *ACM symposium on Applied computing*, Nicosia, Cyprus, 2004.

[BH99] M. Berthold and D. J. Hand. *Intelligent Data Analysis: An Introduction*. Springer, Berlin, 1999.

[BMUT97] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *ACM SIGMOD International Conference on Management of Data*, pages 255–264, Tucson, Arizona, USA, 1997.

[BRS97] S. Brin, M. Rajeev, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *ACM SIGMOD International Conference on Management of Data*, pages 13–15, Tucson, Arizona, USA, 1997.

[BT00] D. W. Bradley and A. M. Tyrrell. Immunotronics : Hardware fault tolerance inspired by the immune system. In *the 3rd International Conference on Evolvable Systems (ICES 2000)*, Lecture Notes in Computer Science, Volume 1801, pages 11–20, Edinburgh, Scotland, UK, 2000. Springer-Verlag.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, Addison-Wesley, 1999.

[CB91] P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *the 6th European Working Session on Learning (EWSL '91)*, Berlin, Germany, 1991. Springer.

[CC] A. Cardoso-Cachopo. Datasets for single-label text categorization. Available at http://www.gia.ist.utl.pt/∼acardoso/datasets/.

[Ces90] B Cestnik. Estimating probabilities: A crucial task in machine learning. In *European Conference on Artificial Intelligence (ECAI 1990)*, pages 147–149, Stockholm, Sweden, 1990.

[CGR91] G. A. Carpenter, S. Grossberg, and D. B. Rosen. Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4:759–771, 1991.

[CHY96] M. S. Chen, J. Han, and P. S. Yu. Data mining: An overview from a database perspective. *IEEE Trans. On Knowledge And Data Engineering*, 8:866–883, 1996.

[CM98] V. Cherkassky and P. Mulier. *Learning from Data: Concepts, Theory, and Methods*. John Wiley, New York, 1998.

[Coh95] W. W. Cohen. Fast effective rule induction. In *the 12th International Conference (ICML-95)*, pages 115–123. Morgan Kaufman, 1995.

[Cri] L. Cristofor. Artool project. Available at http://www.cs.umb.edu/∼laur/ARtool/.

[CT02] H. Cherfi and Y. Toussaint. How far association rules and statistical indices help structure terminology? In *Natural Language Processing and Machine Learning for Ontology Engineering OLT'02*, pages 5–9, Lyon, France, 2002.

[Das99] D. Dasgupta. Immunity-based intrusion detection systems: A general framework. In *the 22nd National Information Systems Security Conference (NISSC)*, pages 18–21, Arlington, Virginia, USA, 1999.

[DCH04] T. D. Do, K. Chang, and S. C. Hui. Web mining for cyber monitoring and filtering. In *2004 IEEE Conference on Cybernetics and Intelligent Systems*, pages 399–404, Singapore, 2004.

[dCT02] L. N. de Castro and J. Timmis. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer-Verlag, 2002.

[dCT03] L. N. de Castro and J. I. Timmis. Artificial immune systems as a novel soft computing paradigm. *Soft Computing*, 7(8):526–544, 2003.

[dCVZ99] L. N. de Castro and F. J. Von Zuben. Artificial immune systems: Part I - basic theory and applications. Technical report, Technical Report - RT DCA 01/99, 1999.

[dCVZ00a] L. N. de Castro and F. J. Von Zuben. The clonal selection algorithm with engineering applications. In *the Workshop on Artificial Immune Systems and Their Applications (GECCO'00)*, pages 36–37, Las Vegas, Nevada, USA, 2000.

[dCVZ00b] L. N. de Castro and F. J. Von Zuben. An evolutionary immune network for data clustering. In *Brazilian Simposium on Artificial Neural Networks (IEEE SBRN'00)*, pages 84–89, 2000.

[dCVZ02] L. N. de Castro and F. J. Von Zuben. Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, 6(3):239–251, 2002.

[DF99] D. Dasgupta and S. Forrest. An anomaly detection algorithm inspired by the immune system. In D. Dasgupta, editor, *Artificial Immune Systems and Their Applications*, pages 262–277. Springer-Verlag, New York, 1999.

[DHF05] T. D. Do, S. C. Hui, and A. C. M. Fong. Prediction confidence for associative classification. In *fourth International Conference on Machine Learning and Cybernetics (ICMLC 2005)*, pages 1993–1998, Guangzhou, China, 2005.

[DS04] F. Debole and F. Sebastiani. An analysis of the relative hardness of reuters-21578 subsets. In *LREC-04, 4th International Conference on Language Resources and Evaluation*, pages 971–974, Lisbon, 2004.

[Dua03] S. Duanmu. *Measuring the Association Rules in Data Mining.* M.Sc. thesis, North Dakota State University, 2003.

[Dun03] M. H. Dunham. *Data Mining - Introductory and Advanced Topics.* Prentice Hall, 2003.

[DZ03] D. Dasgupta and J. Zhou. Reviewing the development of ais in last five years. In *2003 IEEE Congress on Evolutionary Computation*, Canberra, Australia, 2003.

[Fel98] C. Fellbaum. *Wordnet, an Electronic Lexical Database.* MIT Press, 1998.

[FH97] R. Feldman and H. Hirsh. Exploiting background information in knowledge discovery from text. *Intelligent Information Systems*, 9(1):83–97, 1997.

[FPAC94] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri. Self - nonself discrimination in a computer. In *IEEE Symposium on Research in Security and Privacy*, pages 202–212, Oakland, CA, 1994.

[Fre98] J. E. Freund. *Modern Elementary Statistics.* Prentice-Hall, 1998.

[Fre00] A. A. Freitas. Understanding the crucial differences between classification and discovery of association rules: A position paper. *ACM SIGKDD Explorations Newsletter*, 2(1):65–69, 2000.

[FU96] U. Fayyad and R. Uthurusamy. Data mining and knowledge discovery in databases. *Communications of the ACM*, 39(11), 1996.

[Fur99] J. Furnkranz. Exploiting structural information for text classification. *Intelligent Data Analysis*, pages 487–498, 1999.

[Goe] B. Goethals. Frequent pattern mining implementations. Available at http://www.adrem.ua.ac.be/∼goethals/software/.

[Goe02] B. Goethals. *Efficient Frequent Pattern Mining*. Phd thesis, University of Limburg, 2002.

[Goe04] B. Goethals. Memory issues in frequent itemset mining. In *ACM Symposium on Applied Computing (SAC'04)*, pages 530–534, Nicosia, Cyprus, 2004. ACM.

[HC02] J. Han and K. Chang. Data mining for web intelligence. *IEEE Computer*, 35(11):64–70, 2002.

[Heg03] M. Hegland. Algorithms for association rules. *Lecture Notes In Artificial Intelligence*, LNAI2600:226–234, 2003.

[HGN00] J. Hipp, U. Gntzer, and G. Nakhaeizadeh. Algorithms for association rule mining - a general survey and comparison. *ACM SIGKDD Explorations*, 2(1):58–64, 2000.

[HK01] J. Han and M. Kamber. *Data Mining - Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.

[HKK97] E. H. Han, G. Karypis, and V. Kumar. Clustering based on association rule hypergraphs. In *The SIGMOD'97 Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 9–13, Tucson, Arizona, 1997. ACM.

[HL99] E. Hovy and C. Y. Lin. Automated text summarization in summarist. *Advances in Automatic Text Summarization*, pages 81–94, 1999.

[Hol92] J. H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, 1992.

[HP00] J. Han and J. Pei. Mining frequent patterns by pattern-growth: Methodology and implications. *SIGKDD Explorations*, 2(2):14–20, 2000.

[HPY00] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12, Dallas, Texas, USA, 2000.

[HSM01] D. J. Hand, P. Smyth, and H. Mannila. *Principles of Data Mining*. MIT Press, 2001.

[Its01] J. Itskevitch. *Automatic Hierarchical E-Mail Classification Using Association Rules*. M.Sc. thesis, Computing Science, Simon Fraser University, 2001.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[Jol86] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[Kan03] M. Kantardzic. *Data Mining: Concepts, Models, Methods, and Algorithms*. Wiley-IEEE Press, 2003.

[KE95] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks*, pages 1942–1948, Perth, Australia, 1995.

[KJ04] P. Kolari and A. Joshi. Web mining: Research and practice. *Computing in Science and Engineering*, 6(4):49–53, 2004.

[KJL+94] R. Kohavi, G. John, R. Long, D. Manley, and K. Pfleger. MLC++: A machine learning library in C++. *In Tools with Artificial Intelligence*, pages 740–743, 1994.

[KMR+94] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. In B. K. Bhargava N. R. Adam and Y. Yesha, editors, *the 3rd International Conference on Information and Knowledge Management (CIKM'94)*, pages 401–407, Gaithersburg, Maryland, 1994.

[KN99] K. Krishnakumar and J. Neidhoefer. Immunized adaptive critic for an autonomous aircraft control application. In *Artificial Immune Systems and Their Applications*, pages 221–241. Springer-Verlag, 1999.

[Koh95] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, Berlin, 1995.

[KS96] D. Koller and M. Sahami. Toward optimal feature selection. In *International Conference on Machine Learning (ICML'96)*, pages 284–292, Bari, Italy, 1996.

[LHF02] P. Y. Lee, S. C. Hui, and A. C. M. Fong. Neural networks for web information filtering. *Intelligent Systems, IEEE*, 17(5):48–57, 2002.

[LHM98] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *the 4th International Conference on Knowledge Discovery and Data Mining*, pages 80–86, New York, 1998.

[LHP01] W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *2001 IEEE International Conference on Data Mining (ICDM '01)*, pages 369 – 376, San Jose, California, USA, 2001.

[Li01] W. Li. *Classification Based on Multiple Association Rules*. M.Sc. thesis, Thesis, Simon Fraser University, 2001.

[LM98] H. Liu and H. Motoda. *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Kluwer Academic Publishers, 1998.

[LMW00] B. Liu, Y. Ma, and C. K. Wong. Improving an association rule based classifier. In *the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD-2000)*, Lyon, France, 2000.

[LMW01] B. Liu, Y. Ma, and C. K. Wong. Classification using association rules: Weaknesses and enhancements. In Robert L. Grossman, Chandrika Kamath, Philip Kegelmeyer, Vipin Kumar, and Raju R. Namburu, editors, *Data Mining for Scientific Applications*. 2001.

[LMWY03] B. Liu, Y. Ma, C. K. Wong, and P. S. Yu. Scoring the data using association rules. *Applied Intelligence*, 18(2):119–135, 2003.

[LMY03] H. Liu, M. Motoda, and L. Yu. Feature extraction, selection, and construction. In N. Ye, editor, *The Handbook of Data Mining*, pages 409–423. Lawrence Erlbaum Associates, Inc. Publishers, 2003.

[LNHP99] L. V. S. Lakshmanan, R. Ng, J. Han, and A. Pang. Optimization of constrained frequent set queries with 2-variable constraints. In *1999 ACM SIGMOD International Conference on Management of Data*, pages 157–168, Philadelphia, Pennsylvania, United States, 1999.

[MHB+97] J. Moore, E. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, and B. Mobasher. Web page categorization and feature selection using association rule and principal component clustering. In *Workshop on Information Technologies and Systems*, 1997.

[Mic80] R. S. Michalski. Pattern recognition as rule-guided induction inference. *IEEE Transaction On Pattern Analysis and Machine Intelligence*, 2:349–361, 1980.

[Mla98] D. Mladenic. Feature subset selection in text-learning. In *the 10th European Conference on Machine Learning*, pages 95–100, Dorint-Parkhotel, Chemnitz, Germany, 1998.

[MS00] A. Maedche and S. Staab. Discovering conceptual relations from text. In W. Horn, editor, *the 14th European Conference on Artificial Intellignece (ECAI'2000)*, pages 21–25, Berlin, 2000.

[MS01] A. Maedche and S. Staab. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72 – 79, 2001.

[MTV93] H. Mannila, H. Toivonen, and A. I. Verkamo. Improved methods for finding association rules. Technical report, Technical Report TR C-1993-65, Departement of Computer Science University of Helsinki, 1993.

[NCR04] O. Nasraoui, C. Cardona, and C. Rojas. Single pass mining of evolving trends in web data with explicit retrieval similarity measures. In *International World Wide Web Conference*, New York, 2004.

[NCRG03] O. Nasraoui, C. Cardona, C. Rojas, and F. Gonzlez. Tecno-streams: Tracking evolving clusters in noisy data streams with a scalable immune system learning model. In *the 3rd IEEE International Conference on Data Mining (ICDM'03)*, pages 235–242, Melbourne, Florida, USA, 2003.

[NGC+03] O. Nasraoui, F. Gonzalez, C. Cardona, C. Rojas, and D. Dasgupta. A scalable artificial immune system model for dynamic unsupervised learning. In *Genetic and*

Evolutionary Computation Conference (GECCO 2003), pages 219–230, Chicago, IL, 2003.

[NHBM98] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases. Technical report, University of California, Department of Information and Computer Science, 1998.

[Nib87] T. Niblett. Constructing decision trees in noisy domains. In *the 2nd European Working Session on Learning (EWSL)*, pages 67–78, Yugoslavia, 1987.

[N.K73] Jerne N.K. The immune system. *Scientific American*, 229(1):52–60, 1973.

[NK02] O. Nasraoui and R. Krishnapuram. A new evolutionary approach to web usage and context sensitive associations mining. *International Journal of Computational Intelligence and Applications*, 2(3):339–348, 2002.

[NLHP98] R. T. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained association rules. In *1998 ACM SIGMOD International Conference Management of Data*, pages 13–24, Seattle, Washington, USA, 1998.

[NN01] T. Nasukawa and T. Nagano. Text analysis and knowledge mining system. *IBM Systems Journal*, 40(4):967 – 984, 2001.

[PDJ88] M. Potter and K. De Jong. The coevolution of antibodies for concept learning. In *the 5th International Conference on Parallel Problem Solving from Nature (PPSN V)*, volume Springer-Verlag, pages 530–540, Amsterdam, Holland, 1988.

[Pea] PearlSoftware. Cyber snoop parental control software for internet filtering and monitoring. Available at http://www.cyber-snoop.com/.

[PHL01] J. Pei, J. Han, and L. V. S. Lakshmanan. Mining frequent item sets with convertible constraints. In *the 17th International Conference on Data Engineering (ICDE)*, pages 433–442, Heidelberg, Germany, 2001.

[QCJ93] J. R. Quinlan and R. M. Cameron-Jones. Foil: A midterm report. In P. Brazdil, editor, *the 6th European Conference on Machine Learning*, volume 667 of *Lecture Notes in Artificial Intelligence*, pages 3–20, Vienna, Austria, 1993. Springer-Verlag.

[Qui93] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1993.

[RB98] M. Rajman and R. Besancon. Text mining - knowledge extraction from unstructured textual data. In *the sixth Conference of International Federation of Classification Societies (IFCS-98)*, Rome, 1998.

[San04] V. Santosh. *The Random Projection Method*. American Mathematical Society, 2004.

[SB90] E. H. Shortliffe and B. G. Buchanan. A model of inexact reasoning in medicine. pages 259–275, 1990.

[Seb02] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[Sed88] R. Sedgewick. *Algorithms*. Addison-Wesley, second edition, 1988.

[SHS+00] P. Shenoy, J. Haritsa, S. Sudarshan, G. Bhalotia, M. Bawa, and D. Shah. Turbocharging vertical mining of large databases. In *ACM SIGMOD Intl. Conf. on Management of Data*, 2000.

[SON95] A. Savasere, E. Omiecinski, and S. B. Navathe. An efficient algorithm for mining association rules in large databases. In *the 21st International Conference on Very Large Databases (VLDB)*, pages 432–444, ZTrich, Switzerland, 1995.

[SSC97] L. Singh, P. Scheuermann, and B. Chen. Generating association rules from semi-structured documents using an extended concept hierarchy. In *International Conference on Information and Knowledge Management*, pages 193–200, Las Vegas, Nevada, 1997.

[Sur] SurfControl. Cyber patrol - internet filtering software for home, education and business. Available at http://www.cyberpatrol.com/.

[SVA97] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *the 3rd International Conference on Knowledge Discovery in Databases and Data Mining*, pages 67–73, Newport Beach, California, 1997.

[Szi93] E. K. Szig. Advanced engineering mathematics. pages 1155–1158. John Wiley & Sons Inc, seventh edition, 1993.

[TLC99] P. S. M. Tsai, C. C. Lee, and A. L. P. Chen. An efficient approach for incremental association rule mining. In *the 3rd Pacific-Asia Conference, PAKDD-99*, pages 74–83, Beijing, China, 1999.

[TN01] J. Timmis and M. Neal. A resource limited artificial immune system for data analysis. *Knowledge Based Systems*, 14(3-4):121–130, 2001.

[TNH00] J Timmis, M Neal, and J Hunt. An artificial immune system for data analysis. *Biosystems*, 55(1/3):143–150, 2000.

[WB02a] A. B. Watkins and L. C. Boggess. A new classifier based on resource limited artificial immune systems. In *the Congress on Evolutionary Computation, Part of the 2002 IEEE World Congress on Computational Intelligence*, pages 1546–1551, Honolulu, USA, 2002.

[WB02b] A. B. Watkins and L. C. Boggess. A resource limited artificial immune classifier. In *the Congress on Evolutionary Computation, Part of the 2002 IEEE World Congress on Computational Intelligence*, pages 926–931, Honolulu, USA, 2002. IEEE.

[WK91] S. I. Weiss and C. Kulikowski. *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Networks, Machine Learning, and Expert Systems*. Morgan Kaufmann, San Francisco, Calif., 1991.

[WS92] W. J. Wilbur and K. Sirotkin. The automatic identification of stop words. *Journal of Information Science*, 18(1):45–55, 1992.

[Yan03] J. Yang. Classification by association rules: Use minimum set of rules. In *the first Informational Conference on Machine Learning (ICML-2003)*, Piscataway, NJ, 2003.

[YH03] X. Yin and J. Han. CPAR: Classification based on predictive association rules. In *2003 SIAM International Conference on Data Mining (SDM'03)*, San Fransisco, CA, 2003.

[YP97] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *the 14th International Conference on Machine Learning (ICML97)*, pages 412–420, Vanderbilt University, Nashville, TN, 1997.

[ZA02] O. R. Zaiane and M. L. Antonie. Classifying text documents by associating terms with text categories. In *the 13th Australasian Database Conference (ADC'02)*, pages 215–222, Melbourne, Australia, 2002.

[Zak00] M. J. Zaki. Scalable algorithms for association mining. *IEEE TKDE*, 12(3):372–390, 2000.

[ZG03] M. J. Zaki and K. Gouda. Fast vertical mining using diffsets. In *the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, 2003.

[ZKM01] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In *the 7th ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 401–406, San Francisco, CA, 2001.

[ZPOL97] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *the 3rd International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 283–286, Newport, California, 1997.